Examen du 3 novembre 2005

RS : Réseaux et Systèmes

Deuxième année

La notation tiendra compte de la validité des réponses, mais aussi de la présentation et de la clarté de la rédaction. Barème approximatif. Il est conseillé de lire l'intégralité du sujet avant de commencer.

Documents interdits, à l'exception d'une feuille A4 à rendre avec votre copie.

Questions de cours (8pt)

▷ **Question 1.** Définissez les deux fonctions principales des systèmes d'exploitation.

▷ **Question 2.** Décrivez les deux types d'interfaces des systèmes d'exploitation courantes.

▷ **Question 3.** Qu'est ce que l'attente active ? Quels en sont les avantages et inconvéniants ?

▷ **Question 4.** Qu'est ce que la pile d'un processus ? Qu'est ce que le tas ?

▷ **Question 5.** À quoi servent respectivement les appels système `kill`, `fork` et `exec` ?

▷ **Question 6.** Qu'est ce qu'un processus zombie ?

▷ **Question 7.** Que signifie le mot «préemption» ? Donnez deux contextes dans lequel nous avons rencontré ce mot.

▷ **Question 8.** Donnez deux signaux existant sous Unix. Dans quelles conditions sont-ils émis ? Quel est leur effet par défaut sur le processus les recevant ?

▷ **Question 9.** Qu'est ce qu'une condition de compétition ? Dans quelles conditions cela se produit-il ?

▷ **Question 10.** Qu'est ce qu'un interblocage ? Dans quelles conditions cela se produit-il ?

▷ **Question 11.** Qu'est ce qu'une écriture tronquée ? Dans quelles conditions cela se produit-il ?

▷ **Question 12.** Discutez les avantages et inconvénients comparés de write() et fprintf().

▷ **Question 13.** Que se passe-t-il lorsque l'on tape Ctrl-z dans un terminal exécutant une commande comme `emacs` ? Comment inverser l'effet de cette manipulation ?

▷ **Question 14.** Comment prévenir les interblocages ? Donnez et expliquez trois méthodes.

▷ **Question 15.** Pourquoi les signaux ne constituent-ils pas un moyen de communication fiable ? Donnez un scénario mettant en évidence le problème.

▷ **Question 16.** Définissez «interprétation» et «compilation» puis discutez de leurs avantages et inconvénients comparés.

Exercices

Exercice 1 : Clonage de processus (2pt)

Combien de lignes «plop !» imprime chacun des programmes suivants ?

Question 1

```
int main() {
    int i;

    for (i = 0; i < 3; i++)
        fork();
    printf("plop!\n");
    exit(0);
}
```

Question 2

```
void doit() {
    fork();
    fork();
    printf("plop\n");
    fork();
}
int main() {
    doit();
    printf("plop\n");
    exit(0);
}
```

Question 3

```
int main() {
    if (fork() || fork())
        fork();
    printf("plop!\n");
    exit(0);
}
```

Exercice 2 : Édition de liens et symboles multiples (total : 2pt)

Pour chacun des cas ci-après, indiquez si la compilation et l'édition de lien des deux modules sont possibles. Si c'est impossible, indiquez pourquoi. Si les deux sont possibles, indiquez le résultat.

Dans tous les cas, argumentez vos affirmations ; on ne demande pas la valeur des pointeurs affichés, mais ce vers quoi ils pointent.

▷ **Question 1.** ($\frac{1}{4}$pt)

Module A
```
int proc(void) {
```

Module B
```
int proc(void) {
```

▷ **Question 2.** ($\frac{1}{4}$pt)

Module A
```
int proc(void) {
```

Module B
```
int proc(void);
int toto(void) {
  return proc();
}
```

▷ **Question 3.** ($\frac{1}{2}$pt)

Module A
```
int x;
int y;
int carre(int i) {
   x = i;
   y = i * i;
   return y;
}
```

Module B
```
double x ;
int inv(int i) {
  x = 1.0 / (double) i;
  return x;
}
```

▷ **Question 4.** (1pt)

Module A
```
void p2(void);
int main() {
  p2();
  return 0;
}
```

Module B
```
#include <stdio.h>
char main;
void p2() {
  printf("0x%x\n", main);
}
```

Exercice 3 : Relecture de code (3pt)

Pour chacun des extraits de code suivants, déterminez s'ils contiennent des anomalies (erreur de programmation, erreur d'utilisation ou bien code non optimal). Pour chaque anomalie trouvée, donnez sa localisation et une explication.

Question 1
```
1  int i;
2  pid_t pid;
3  /* Creation de NBENFANTS enfants */
4  for (i=0 ; i<NBENFANTS ; i++){
5    pid = fork();
6    switch(pid){
7      case -1:
8        perror(NULL);
9        exit(EXIT_FAILURE);
10     case 0:
11       printf("Père: prêt\n");
12       break;
13     default:
14       printf("Fils: clonage est réussit\n");
15       break;
16   }
17 }
```

Question 2
```
1  void do_mail() {
2    int p[2];
3    pipe(p);
4
5    if (fork()) {
6      dup2(p[1],1);
7      write(p[1],"coucou", 7);
8      close(p[0]); close(p[1]);
9    } else {
10     dup2(p[0], 0);
11     close(p[0]); close(p[1]);
12     execlp("mail", "mail", "bob@localhost");
13   }
14
15   wait(NULL);
16   printf("Message parti\n");
17 }
18
19 main() {
20   while (1) {
21     do_mail();
22     sleep(5);
23   }
24 }
```

Problème : Pirates informatiques (5pt)

L'infâme pirate Barbe-Noire et ses 3 non moins infâmes lieutenants (LeBorgne, Long Jean d'Argent et Rackham le Rose) fêtent avec force rhum la prise d'un magnifique galion espagnol. L'alcool est-il frelaté ? Toujours est-il que chacun des 4 hommes se met à avoir le comportement suivant :

- RÉPÉTER 10 fois
  - Essayer d'entrer dans la cale pour admirer le trésor
  - Admirer le trésor dans la cale pendant T_a secondes
  - Sortir de la cale
  - Cuver son rhum pendant T_c secondes
- FIN RÉPÉTER
- Mourir (l'alcool est vraiment frelaté !)

(les valeurs de T_a et T_c importent peu)

Même ivres, les pirates respectent le code de la piraterie qui stipule que :

- Barbe-Noire doit être seul quand il contemple le trésor :
  - Lorsque Barbe-Noire admire le trésor, si un de ses lieutenants essaye d'entrer dans la cale, il attend que Barbe-Noire en soit sorti avant d'entrer à son tour.
  - Si Barbe-Noire veut entrer dans la cale alors qu'un (ou plusieurs) de ses lieutenant(s) admire(nt) le trésor, Barbe-Noire attend qu'il n'y ait plus personne dans la cale avant d'entrer à son tour.
- Les lieutenants peuvent admirer le trésor ensembles :
  Ainsi si un lieutenant est en train d'admirer le trésor dans la cale et qu'un autre lieutenant essaye d'entrer dans la cale, cet autre lieutenant entre dans la cale sans attendre (sauf si Barbe-Noire attend déjà devant la porte).
- Personne ne double Barbe-Noire :
  Si un ou plusieurs lieutenants admirent le trésor et que Barbe-Noire attend d'entrer dans la cale, les autres lieutenants qui auraient envie d'admirer le trésort doivent attendre que Barbe-Noire soit entré dans la cale, ait admiré le trésor et soit sorti de la cale avant d'entrer à leur tour.

▷ **Question 1.** De quel schéma de syncronisation ce problème se rapproche-t-il ?

▷ **Question 2.** Implémentez les processus `BarbeNoire` et `Lieutenant` grâce à des sémaphores.