Faculté d'Electronique et Informatique 07/05/09
Licence ACAD - S3 – 2008/2009
Module Système d'exploitation

EMD

(Durée : 1h30)

Questions :

1) (3 pts) Quelles sont les tâches de contrôles qui font appel au scheduler (lancer un nouveau processus) et dans quel cas ?

Nom Tâche de contrôle	Cas d'appel
1.	
2. ……	

2) (2 pts) Soit un système à temps partagé où la priorité d'un processus est calculée comme suit :
Priorité = (instant présent – instant d'arrivée)/ (instant d'obtention du dernier quantum).

La priorité la plus grande correspond à la plus petite valeur.

2.1. Quel est l'objectif d'une telle stratégie d'affectation de priorité.
2.2. Donner le contenu d'un PCB.

Exercice 1 (8pts):

A/ On s'intéresse à la pollution de l'air dans un tunnel de circulation de voitures. Un ordinateur de contrôle active dès son démarrage une interruption périodique ***IT1*** qui examine le nombre ***N*** de véhicules entrés dans le tunnel. La période de cette interruption est de 2h. Lorsque l'ordinateur n'a pas de tâche de contrôle à effectuer, il exécute un programme de fond.

1. Quels sont les programmes qui doivent être définis dans ce cas ?
2. Ecrire ces programmes.

B/ Lorsque ***N*** dépasse un certain seuil ***S***, ***IT1*** ne s'exécutera plus et une autre interruption ***IT2*** est activée. ***IT2*** est une interruption périodique qui doit faire des mesures de la qualité de l'air dans le tunnel toutes les heures.

3. On suppose qu'à la deuxième exécution de ***IT1***, ***N*** dépasse le seuil **S.** donner la synoptique d'exécution sur 6h de temps (unité du temps=1h)
4. Donner dans l'ordre de priorité décroissant les interruptions impliquées.
5. Donner le masque d'interruption correspondant à chacune des interruptions (1 :armée, 0 :non armée).
6. Donner le code des routines ***IT1*** et ***IT2***.

Exercice 2 (7 pts):

Soit le programme suivant :

```
#include <unistd.h>
#include <sys/types.h>
#include <stdlib.h>
#include <stdio.h>
#include <signal.h>

#define val ...

int compteur, nb1 = 1, nb2 = 1;
pid_t pid1, pid2;
void Hand_P (int sig) {
      if (compteur == val) {
            kill(pid1, SIGINT);
            kill(pid2, SIGINT);
            exit (0);
      };
      compteur++;
      if (compteur % 2 == 1) kill(pid2, SIGUSR2); /* % :op. modulo */
      else kill(pid1, SIGUSR1);
}
void Hand_F1 (int sig) {
      printf("Fils1: %d\n", nb1);
      nb1++;
      kill(getppid(), SIGCONT);
}
void Hand_F2 (int sig) {
      printf("Fils2: %d\n", nb2*nb2);
      nb2++;
      kill(getppid(), SIGCONT);
}
int main() {
      compteur = 0;
      signal (SIGCONT, Hand_P);
      signal (SIGUSR1, Hand_F1);
      signal (SIGUSR2, Hand_F2);
      pid1 = fork();
      if(pid1 == 0) {
            printf("Fils1 est pret\n");
            while(1);
      }
      pid2 = fork();
      if(pid2 == 0) {
            printf("Fils2 est pret\n");
            while(1);
      }
      sleep(4);
      kill(pid1, SIGUSR1);
      while(1);
      exit(0);
}
```

1. Donner les affichages réalisés par ce programme pour val = 3, val = 5.
2. Déduire la tâche réalisée par chaque processus fils.
3. Quel problème se pose dans le cas de la suppression des instructions :

```
kill(pid1, SIGINT);
kill(pid2, SIGINT);
```