EISTI	Examen de rattrapage
T. Gherbi - J.A. Lorenzo – S. Yassa	Système d'exploitation
ING1-GI-GM	Année 2017–2018

Modalités

— Durée : 2 heures.

— Vous devez rédiger votre copie à l'aide d'un stylo à encre exclusivement.

— Toutes vos affaires (sacs, vestes, trousse, etc.) doivent être placées à l'avant de la salle.

— Une seule feuille manuscrite (pas de photocopies) est autorisée.

— Aucune question ne peut être posée aux enseignants, posez des hypothèses en cas de doute.

— Aucune machine électronique ne doit se trouver sur vous ou à proximité, même éteinte.

— Aucune sortie n'est autorisée avant une durée incompressible d'une heure.

— Aucun déplacement n'est autorisé.

— Aucun échange, de quelque nature que ce soit, n'est possible.

Exercice 1 : Programmation de processus (5 points = 3 + 2)

1. Écrivez un programme C qui crée 2 fils. Chaque processus doit afficher son PID à l'écran. Le père doit attendre la fin du 1er fils. Lorsque le 1er fils se termine, il envoie un code de retour, qui doit être récupéré par le père et affiché à l'écran.

2. Exécutez le code suivant en imaginant des PID pour les différents processus crées et donnez les messages affichés.

```
#include<unistd.h>

int main() {

        printf("Hello %d\n",getpid());

        fork();

        fork();

        printf("Hi %d:%d\n",getpid(), getppid());

}
```

Exercice 2 : Gestion des fichiers (3 points)

Supposons un nœud d'information (i-node) en Unix (contenant 10 adresses directes et 3 adresses indirectes de blocs) qui représente un fichier f.

Quel est le nombre et le type d'adresses (directe, indirecte de niveau 1, indirecte de niveau 2, indirecte de niveau 3) nécessaires si la taille du fichier f est de 56320 octets, le bloc occupe 512 octets et l'adresse du bloc est donnée sur 16 bits ?

Exercice 3 : Ordonnanceur (5 points)

Soient cinq processus prêts A, B, C, D et E ; tel que : A arrive en premier, B arrive 2 unités de temps après A, C arrive 1 unité de temps après B, D arrive 2 unités de temps après C et E arrive 3 unités de temps après D. Les temps nécessaires pour l'exécution des processus A, B, C, D et E sont respectivement 8, 4, 2, 5 et 3 unités de temps. Le temps de commutation est supposé nul.

Processus	Temps d'exécution	Temps d'arrivée
A	8	0
B	4	2
C	2	3
D	5	5
E	3	8

Représentez sur un axe de temps horizontal, l'exécution de ces processus puis calculez :

— le temps de séjour de chaque processus.

— le temps moyen de séjour.

— le temps d'attente : temps de séjour - temps d'exécution du travail.

— le temps moyen d'attente.

— le nombre de changements de contexte (Note : un changement de contexte se produit à chaque fois qu'un processus acquiert le processeur)

en utilisant les techniques :

1. FCFS (First Come First Served)
2. SFJ (Shortest Job First)

Exercice 4 : Mémoire virtuelle (3 points)

Supposons une machine avec 32 kOctets de mémoire, une plage d'adressage virtuelle de 16 bits et 8 kOctets de taille de page.

1. Dans l'adresse virtuelle, combien de bits faut-il utiliser pour le numéro de page virtuelle ? Quelle est la taille maximale de mémoire qui peut être gérée ? Justifiez.
2. Traduisez l'adresse virtuelle suivante en adresse réelle :

   0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0

   en utilisant la table de pages suivante :

	Physical address	Valid bit
7	00	0
6	00	0
5	11	0
4	00	0
3	01	1
2	00	0
1	10	1
0	00	1

3. Une fois l'adresse trouvée en mémoire physique, faudra-t-il remplacer la page ? Pourquoi ?

Questions courtes (4 points)

1. Quels sont les avantages et inconvénients de la méthode d'allocation contiguë de la mémoire secondaire ? A quel type de média est-elle adaptée?

2. Où se situe le MBR et que contient-il?

3. Quelle est la différence entre apt-get et apt-get upgrade?

4. Qu'est-ce qu'un processus orphelin et qu'est-ce qu'un processus zombie?

GL3

Exercice 1 : Programmation de processus (5 points = 3 + 2)

1. Écrivez un programme C qui crée 2 fils. Chaque processus doit afficher son PID à l'écran. Le père doit attendre la fin du 1er fils. Lorsque le 1er fils se termine, il envoie un code de retour, qui doit être récupéré par le père et affiché à l'écran.

2. Exécutez le code suivant en imaginant des PID pour les différents processus crées et donnez les messages affichés.

```
#include<unistd.h>

int main()
{ printf("Hello %d\n",getpid());
  fork();
  fork();
  printf("Hi %d:%d\n",getpid(), getppid());
}
```

1.
```
int main(int argc, char** argv){
    int code;
    int status;
    int i;
    for(i=0; i<2; i++){
        pid_t pid = fork();
        if (pid == -1){
            printf("erreur");
            exit(1);
        }
        else if (pid == 0){
            code = wait(&status);
            printf("PID: %i, PPID: %i, code: %i, getpid(), getppid, code);
        } else{
            printf("PID: %i, PID du fils: %i", getpid(), pid);
```

2. Sortie

Hello 15633

Hi 16325:16326

GU3

Exercice 2 : Gestion des fichiers (3 points)

Supposons un nœud d'information (i-node) en Unix (contenant 10 adresses directes et 3 adresses indirectes de blocs) qui représente un fichier f.

Quel est le nombre et le type d'adresses (directe, indirecte de niveau 1, indirecte de niveau 2, indirecte de niveau 3) nécessaires si la taille du fichier f est de 56320 octets, le bloc occupe 512 octets et l'adresse du bloc est donnée sur 16 bits ?

f : 56 320 o

bloc : 512 o

adresse : 16 b → 2 o

56 320 % 512 = 110

Il y a 110 blocs dans le fichier

↳ 10 adresses directes, il reste 100 blocs

512 / 2 = 256 numéros de blocs dans un bloc

Comme 100 < 256, les 100 dernières adresses sont indirectes de niveau 1

Au final :

- 10 adresses directes
- 100 adresses indirectes de niveau 1

Exercice 3 : Ordonnanceur (5 points)

Soient cinq processus prêts A, B, C, D et E ; tel que : A arrive en premier, B arrive 2 unités de temps après A, C arrive 1 unité de temps après B, D arrive 2 unités de temps après C et E arrive 3 unités de temps après D. Les temps nécessaires pour l'exécution des processus A, B, C, D et E sont respectivement 8, 4, 2, 5 et 3 unités de temps. Le temps de commutation est supposé nul.

Processus	Temps d'exécution	Temps d'arrivée
A	8	0
B	4	2
C	2	3
D	5	5
E	3	8

Représentez sur un axe de temps horizontal, l'exécution de ces processus puis calculez :
— le temps de séjour de chaque processus.
— le temps moyen de séjour.
— le temps d'attente : temps de séjour - temps d'exécution du travail.
— le temps moyen d'attente.
— le nombre de changements de contexte (Note : un changement de contexte se produit à chaque fois qu'un processus acquiert le processeur)

en utilisant les techniques :
1. FCFS (First Come First Served)
2. SFJ (Shortest Job First)

1) FCFS : temps de séjour : A → 8

B → 8+4-2 = 10

C → 14-3 = 11

D → 19-5 = 14

E → 22-8 = 14

temps moyen = $\frac{8+10+11+14+14}{5} = 11,4$

temps attente : A → 0

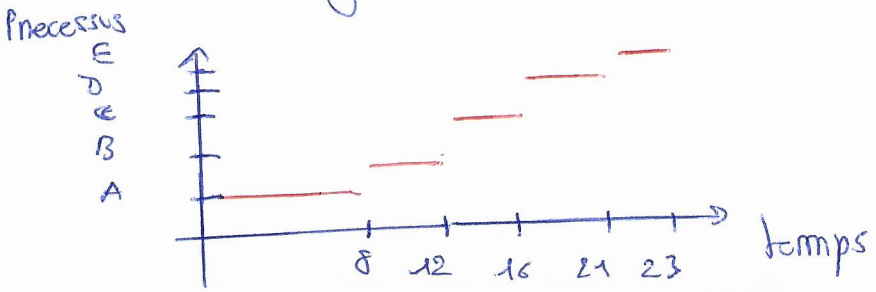
B → 6

C → 9

D → 9

E → 11

temps moyen d'attente = 7



nombre changements de contexte : 4

Q3

2) SFJ temps de séjour : A → 8

C → 8+2-3 = 7

E → 13-8 = 5

B → 17-2 = 15

D → 22-5 = 17

temps moyen de séjour = $\frac{8+7+5+15+17}{5} = 10,4$

temps d'attente : A → 0

B →

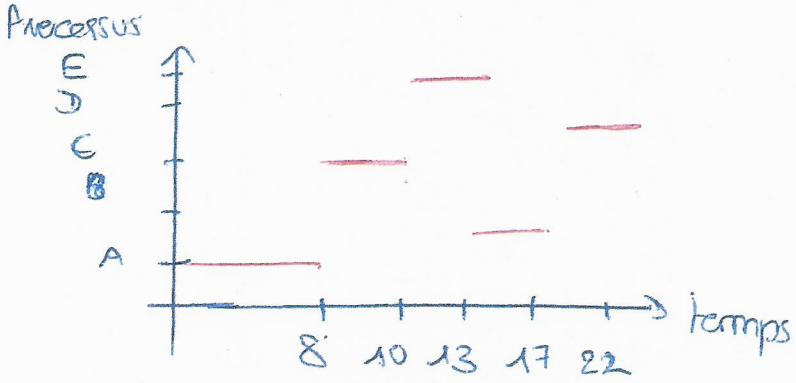
C → 5

E → 2

B → 11

D → 12

temps moyen d'attente = 6



nombre de changement de contexte : 4

Exercice 4 : Mémoire virtuelle (3 points)
Supposons une machine avec 32 kOctets de mémoire, une plage d'adressage virtuelle de 16 bits et 8 kOctets de taille de page.

1. Dans l'adresse virtuelle, combien de bits faut-il utiliser pour le numéro de page virtuelle ? Quelle est la taille maximale de mémoire qui peut être gérée ? Justifiez.
2. Traduisez l'adresse virtuelle suivante en adresse réelle :
   0 1 1 0 0 1 0 1 0 0 1 0 0 0 1 0

en utilisant la table de pages suivante :

8 bits de mémoire physique

	Physical address	Valid bit
7	00	0 X
6	00	0 X
5	11	0 X
4	00	0 X
3	01	1 ✓ ←
2	00	0 X
1	10	1 ✓
0	00	1 ✓

7 6 5 4 3 2 1 0

3. Une fois l'adresse trouvée en mémoire physique, faudra-t-il remplacer la page ? Pourquoi ?

1) Offset = x tq 2^x = 8 Octets ⟺ x = 13 bits

numéro page virtuelle 16 − 13 = 3 bits

Il s'agit de la mémoire physique car la mémoire virtuelle n'est pas gérée il y a donc 15 bits allouées à cette mémoire donc < 32768 octets.

2) num de page virtuelle : 011 sur 3 bits

num cadre de page physique : 2 bits ⇒ 01 le bit de validité est sur 1

traduit : 01 001 0100 0001 0

3) le bit de validité associé à 011 est 1 donc true, la page existe en RAM pas besoin de la changer.

Q13

Questions courtes (4 points)

1. Quels sont les avantages et inconvénients de la méthode d'allocation contiguë de la mémoire secondaire ? A quel type de média est-elle adaptée ?

2. Où se situe le MBR et que contient-il ?

3. Quelle est la différence entre apt-get et apt-get upgrade ?

4. Qu'est-ce qu'un processus orphelin et qu'est-ce qu'un processus zombie ?

1) Avantages: pas de surcharge d'adresses, exécution plus rapide des processus
Inconvénients: on ne peut pas allouer un programme si sa taille dépasse celle d'une partition, on perd beaucoup d'espace de mémoire à cause de la fragmentation
Utile pour les disques de musique

2) Le MBR se situe dans le premier ~~disque~~ secteur de chaque disque dur. Il contient des infos permettant d'identifier l'emplacement et le statut d'un système d'exploitation afin de le charger dans la mémoire principale ou la mémoire vive.

3) apt-get : permet l'installation et la désinstallation de paquets en provenance d'un dépôt APT
apt-get upgrade : met à jour les paquets installés sans les supprimer et sans en installer de nouveaux

4) Processus orphelin : processus qui n'a pas de père
Processus zombie : processus qui s'est arrêté mais qui possède encore un PID