


|   |                               |
|---|-------------------------------|
|  | <b>Examen Session Normale</b> |
| I. Ben Amor, S. Bouzidi, T. Gherbi, J.A. Lorenzo, S.Yassa,                        | Système d'exploitation        |
| ING1-GI-GM  | Année 2020–2021               |

### 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.
- Aucun document n'est autorisé.
- 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.

### QCM à répondre sur votre copie et non pas sur le sujet : (5 points)

- a) Que désigne les lettres **sd** dans `/dev/sda2` ?
- ☐ Le type du bus
  - ☐ La lettre de périphérique
  - ☐ Le numéro de partition
  - ☐ Aucune des réponses n'est valide.
- b) La commande **`docker run -it ubuntu /bin/bash`**
- ☐ Crée une image *ubuntu* sur l'ordinateur.
  - ☐ Télécharge d'abord l'image *ubuntu* si elle n'est pas disponible sur l'ordinateur.
  - ☐ Arrête tous les conteneurs qui utilisent l'image *ubuntu*.
  - ☐ Aucune des réponses n'est valide.
- c) La commande **`apt-get dist-upgrade...`**
- ☐ s'assure qu'en aucun cas des paquets déjà installés ne soient supprimés.
  - ☐ resynchronise un fichier répertoriant les paquets disponibles et sa source.
  - ☐ ajoute une gestion intelligente des changements de dépendances dans les nouvelles versions des paquets.
  - ☐ Aucune des réponses n'est valide.

- d) Les programmes systèmes d'un système d'exploitation...
- ☐ s'exécutent en mode kernel.
  - ☐ s'exécutent en mode utilisateur.
  - ☐ peuvent commuter entre mode kernel et mode utilisateur.
  - ☐ possèdent leur propre mode d'exécution isolé.
- e) Dans une mémoire cache, la tendance à réutiliser des données récemment accédées s'appelle
- ☐ Localité de cache
  - ☐ Localité spatiale
  - ☐ Localité temporelle
  - ☐ Aucune des réponses n'est valide.

### Questions de cours : (5 points)

- a) Quelles sont les avantages et les inconvénients d'une allocation indexée du disque ?
- a) Valgrind est une suite d'outils de profilage et de débogage qui permet de détecter des problèmes de gestion mémoire dans des programmes écrits en C / C++. Donnez une cause possible pour le message d'erreur suivant généré par Valgrind : "Invalid write of size 4".
- b) Expliquez les différences entre les ordonnanceurs préemptifs et non préemptifs.
- c) Expliquez ce qu'est un fichier Dockerfile et donnez un exemple d'utilisation.
- d) À quoi sert le fichier /etc/apt/sources.list ? Expliquez en détail.

### Exercice 1 : Programmation de processus (3 points)

- a) Déterminez le nombre de processus créés par les programmes ci-dessous. Justifier votre réponse.

| Code1   | Code 2   | Code 3  |
|---|--|---|
| <pre>int main() {     fork();     fork();     fork(); }</pre> | <pre>int main() {     if (fork() &gt; 0)         fork(); }</pre> | <pre>int main() {     int cpt=0;     while (cpt &lt; 3) {         if (fork() &gt; 0)             cpt++;         else             cpt=3;     } }</pre> |

b) Précisez l'état final de chaque processus en exécutant le programme ci-dessous avec les arguments 10 20 puis 10 0. Donnez une explication.

```
int main(int argc, char * argv[]) {
    pid_t pid;
    int attente_fils, attente_pere;

    if(argc != 3)
    {
        printf("usage: ex1 'attente_pere' 'attente_fils'\n");
        exit(-1);
    }

    attente_pere = atoi(argv[1]);
    attente_fils = atoi(argv[2]);

    switch(pid=fork()) {
        case -1:
            perror("fork error");
            break;
        case 0:
            sleep(attente_fils);
            printf("fils attente finie\n");
            break;
        default:
            sleep(attente_pere);
            printf("pere attente finie\n");
            break;
    }
    return 0;
}
```

## Exercice 2 : Le système de fichiers (2 points)

On considère un système de fichiers similaire à celui d'Unix dans lequel l'information concernant les blocs de données de chaque fichier est accessible à partir du i-noeud.

On suppose que :

- Le système de fichiers utilise des blocs de données de taille fixe 1K (1024 octets).
- L'i-noeud de chaque fichier (ou répertoire) contient 14 pointeurs directs sur des blocs de données, 1 pointeur indirect simple, 1 pointeur indirect double et 1 pointeur indirect triple.
- Chaque pointeur (numéro de bloc) est représenté sur 4 octets.

a) Quelle est la plus grande taille de fichier que peut supporter ce système de fichiers?

b) On considère un fichier contenant 200 000 octets. Combien de blocs de données sont-ils nécessaires (au total) pour représenter ce fichier sur le disque ?

### Exercice 3 : Ordonnancement (2 points)

Soient cinq processus prêts A, B, C, D et E, avec les temps d'arrivée au système et les temps d'exécution indiqués dans le tableau ci-dessous :

| Processus | Temps d'exécution | Temps d'arrivée |
|-----------|-------------------|-----------------|
| A         | 3                 | 0               |
| B         | 4                 | 0               |
| C         | 1                 | 1               |
| D         | 1                 | 4               |
| E         | 2                 | 5               |

En supposant que le temps de commutation est nul, calculer :

- le temps de séjour de chaque processus.
- le temps moyen de séjour.
- le temps d'attente : temps de séjour - temps d'exécution.
- le temps moyen d'attente.
- le nombre de changements de contexte

en utilisant les techniques :

1. *Shortest Job First*.
2. *Shortest Remaining Time*.

### Exercice 4 : Mémoire virtuelle (3 points)

Supposons que nous avons une machine ayant 2 GBytes de mémoire physique et une plage d'adressage virtuelle de 32 bits avec 1024 pages. Répondez aux questions suivantes :

1. Quelle est la taille d'une page ?
2. Combien de cadres de page (frames) y-a-t-il dans la mémoire physique ?
3. Supposons deux pages A et B d'un processus qui sont affectées par la table de pages vers le même cadre de page en mémoire physique. Si la séquence d'accès par le processus est A, B, A, quelle sera la valeur du valid-bit de la page A à la fin de cette séquence ? Justifiez.

# Corrigé de l'examen

## QCM : (5 points)

- a) Que désigne les lettres **sd** dans `/dev/sda2`
  - ☐ Le type du bus
- b) La commande **docker run -it ubuntu /bin/bash**
  - ☐ Téléchargera d'abord l'image *ubuntu* si elle n'est pas disponible sur l'ordinateur.
- c) La commande **apt-get dist-upgrade...**
  - ☐ ajoute une gestion intelligente des changements de dépendances dans les nouvelles versions des paquets.
- d) Les programmes systèmes d'un système d'exploitation...
  - ☐ s'exécutent en mode kernel.
- e) Dans une mémoire cache, la tendance à réutiliser des données récemment accédées s'appelle
  - ☐ Localité temporelle

## Questions de cours : (5 points)

- a) Quelles sont les avantages et les inconvénients d'une allocation indexée du disque ?  
Avantages : facilite les modifications et l'accroissement. Accès aléatoire.  
Inconvénients : les index prennent de l'espace. Taille du fichier limitée.
- b) Expliquez une possible cause pour le message suivant de Valgrind : "Invalid write of size 4".  
Essai d'écriture d'un type Integer sur une position de mémoire qui n'a pas été réservée par le programme.
- c) Expliquez la différence entre les ordonnanceurs préemptif et non préemptif.
- d) Expliquez ce qu'est un fichier Dockerfile et donner un exemple de comment l'utiliser.
- e) À quoi sert le fichier `/etc/apt/sources.list` ? Expliquez en détail.

## Exercice 1 : Programmation de processus (3 points)

A)

Code 1 : 8 processus sont créés

Code 2: 3 processus sont créés

Code 3: 4 processus sont créés

B)

1- Le père meurt avant son fils, le fils devient orphelin

2- Le fils meurt avant son père, le père est en sommeil, il ne lit pas le code de retour de son fils. Le fils devient zombie.

## Exercice 2 : Le système de fichiers (2 points)

- a) on a chaque pointeur (numéro de bloc) est représenté sur 4 octets. Donc chaque bloc de données peut contenir 256 pointeurs (car  $1K/4$ )

( $14 * 1k$ ) pour direct, ( $256 * 1K$ ) pour indirect simple, ( $256 * 256 * 1K$ ) pour indirect double, ( $256 * 256 * 256 * 1K$ ) pour indirect Triple.

La plus grande taille de fichier est :

$$\begin{aligned}\text{Taille maximale} &= (14 * 1K) + (256 * 1K) + (256 * 256 * 1K) + (256 * 256 * 256 * 1K) \\ &= (14 + 256 + 256*256 + 256*256*256) K \\ &= 16\,843\,022\, K \text{ octets} \sim 16\, Go\end{aligned}$$

- b) Pour un fichier contenant 200 000 octets, on remarque que :

$$200\,000 = 195 * 1024 + 320$$

Il faudra donc 196 blocs pour conserver les données de ce fichier.

L'i-noeud ne dispose que de 14 blocs directs ( $196 - 14 = 182$ ), il va donc falloir utiliser les pointeurs indirects pour conserver le reste des données. Le pointeur indirect simple du i-noeud est suffisant puisque 182 est inférieur à 256.

Il faudra donc au total 197 blocs sur le disque (196 pour les données + 1 pour le bloc de pointeurs indirects)

## Exercice 3 : Gestion de processus - ordonnancement (2 points)

### 1. SJF

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|
| A | A | A | C | D | E | E | B | B | B | B  |    |    |    |    |    |    |    |    |

A arrive

D arrive E arrive

B arrive

C arrive

| Processus | Temps séjour  | Temps d'attente |
|-----------|---------------|-----------------|
| A         | $3 - 0 = 3$   | $3 - 3 = 0$     |
| B         | $11 - 0 = 11$ | $11 - 4 = 7$    |
| C         | $4 - 1 = 3$   | $3 - 1 = 2$     |
| D         | $5 - 4 = 1$   | $1 - 1 = 0$     |
| E         | $7 - 5 = 2$   | $2 - 2 = 0$     |

$$T_{\text{moyen\_sejour}} = (3+11+3+1+2) / 5 = 4 \text{ unités de temps}$$

$T_{\text{moyen\_attente}} = (0+7+2+0+0) / 5 = 1,8$  unités de temps  
 5 changements de contexte (en comptant le changement initial)

### 1. SRT

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|
| A | C | A | A | D | E | E | B | B | B | B  |    |    |    |    |    |    |    |    |

A arrive  
 B arrive  
 C arrive  
 D arrive E arrive

| Processus | Temps séjour  | Temps d'attente |
|-----------|---------------|-----------------|
| A         | $4 - 0 = 4$   | $4 - 3 = 1$     |
| B         | $11 - 0 = 11$ | $11 - 4 = 7$    |
| C         | $2 - 1 = 1$   | $1 - 1 = 0$     |
| D         | $5 - 4 = 1$   | $1 - 1 = 0$     |
| E         | $7 - 5 = 2$   | $2 - 2 = 0$     |

$T_{\text{moyen\_sejour}} = (4+11+1+1+2) / 5 = 3,8$  unités de temps  
 $T_{\text{moyen\_attente}} = (1+7+0+0+0) / 5 = 1,6$  unités de temps  
 6 changements de contexte (en comptant le changement initial)

### Exercice 4 : Mémoire virtuelle (3 points)

Supposons une machine avec 2 GBytes de mémoire physique et une plage d'adressage virtuelle de 32 bits avec 1024 pages. Répondre les questions suivantes :

4. Quelle est la taille d'une page ?

$1024 \text{ pages} = 2^{10} \text{ pages}$ , donc 10 bits utilisés pour le numéro de page.

$32 - 10 = 22$  bits pour l'offset. Donc  $2^{22} = 4 \text{ Mbytes} / \text{page}$ .

5. Combien de cadres de page y a-t-il dans la mémoire physique ?

$2 \text{ GBytes} = 2048 \text{ MBytes}$

$2048 \text{ MBytes} / 4 \text{ MB/page} = 512 \text{ cadres de page}$

6. Supposons deux pages A et B d'un processus qui sont affectées par la table de pages vers le même cadre de page en mémoire physique. Si la séquence d'accès par le processus est

A, B, A, quelle sera la valeur du valid-bit de la page B à la fin de cette séquence ? Justifiez.

1. Accès à A. Valid-bit A = 1. Valid-bit B = 0
2. Accès à B. Valid-bit A = 0. Valid-bit B = 1
3. Accès à A. Valid-bit A = 1. Valid-bit B = 0.