

Système 1
Examen du 02/06/2021

Vous avez 2h. Vous avez le droit à 1 feuille recto-verso avec vos notes de cours. Le barème est donné à titre indicatif, il peut évoluer.

Exercice 1 _____ **Gestion de la mémoire (4 points)**

Le système étudié :

- Adresse sur 32 bits
- Blocs de 256 octets
- UNIX (gestion des fichiers par i-nodes)
- chaque i-node possède 4 adresses directes et 3 niveaux d'indirection (1 adresse par niveau).

$$\text{bloc } 256 \text{ o } \text{adr } 40 \text{ 1 bloc peut contenir } \frac{2^8}{2^2} = 2^6 \text{ adresses, } 64 \text{ adresses}$$

1. Combien d'adresses un bloc peut-il contenir ? *Donnez le calcul et le résultat.*
2. Quelle est la taille maximale d'un fichier géré par un i-node sans indirection ? *Donnez le calcul et le résultat.*
3. Quelle est la taille maximale d'un fichier décrit par un i-node ? *Donnez le calcul et le résultat.*
4. Donnez une description du i-node et des blocs d'indirection utilisés pour un fichier contenu sur 6 blocs, aux adresses 0xFF00, 0xFF04, 0xFE42, 0xFFA0, 0xFF21 et 0xFF02.

Exercice 2 _____ **FAT 32 (3 points)**

Le système étudié :

- Adresse sur 32 bits
- Blocs de 64 octets
- gestion des fichiers par fat

1. Quelle est la taille maximale de votre système ? *Donnez le calcul et le résultat.*
2. Donnez une description de la FAT pour un fichier contenu sur 6 blocs, aux adresses 0xFF00, 0xFF04, 0xFE42, 0xFFA0, 0xFF21 et 0xFF02.

Exercice 3 _____ **Allocation à la demande (3 points)**

On dispose d'un système monoprocesseur en multilprogrammation doté d'une mémoire virtuelle paginée. Une adresse virtuelle a 32 bits dont les 12 bits de poids faibles encodent le déplacement dans la page, les 20 autres le numéro de la page virtuelle.

À chaque processus est associé une table des pages dont la i-ème entrée décrit la i-ème page virtuelle du processus :

bit 31	bit 30	bit 29	bit 28	bit 20	bit 19	bit 0
présence	modifiée	accédée		protection		num de cadre

- présence vaut 1 si la page est présente en mémoire centrale, 0 sinon.
- modifiée vaut 1 si la page a été modifiée depuis son chargement, 0 sinon.
- accédée vaut 1 si la page a été référencée depuis un certain temps.
- protection indique les droits d'accès du processus à la page.

- num de cadre indique le numéro du cadre de la mémoire physique où se trouve la page virtuelle (si elle est présente en mémoire centrale).

Dans le contexte de chaque processus, un registre repère le début de la table des pages.

1. Décrivez sous forme algorithmique les opérations réalisées lors du décodage d'une adresse (envisagez selon la valeur du bit de présence). *La table des pages d'un processus est dans la mémoire centrale.*

Le décodage d'une adresse virtuelle consiste à aller chercher la page physique correspondante.

2. Ici quelle est la taille d'une page (en Ko) et combien de pages la table contient-elle ? Combien de cadre la mémoire centrale contient-elle ? quelle est a priori la taille de la mémoire centrale ?

Exercice 4 Algo de remplacement (6 points)

Votre système possède 4 cadres et 8 pages. Les demandes d'accès mémoires sont les suivantes :

instant	0	3	10	11	15	18	21	30	33	38	39	44	51
page demandée	3	4	0	7	5	7	1	0	3	0	7	4	1

1. Donnez le déroulé de l'algorithme FIFO ainsi que son nombre de défaut de pages.
2. Donnez le déroulé de l'algorithme LFU (least frequently used) ainsi que son nombre de défaut de pages.
3. Donnez le déroulé de l'algorithme d'horloge ainsi que son nombre de défaut de pages.

Pour le déroulé d'un algorithme, donnez (au moins) instant par instant l'occupation des cadres. Pour la clarté de vos présentations, vous pouvez supposer qu'une page présente dans un cadre peut se déplacer dans un autre cadre avec un coût nul.

Exercice 5 Fork (5 points)

On dispose du programme main.c suivant :

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
int main(int argc, char* argv[]){
    int res[3];
    res[0] = fork();
    res[1] = fork();
    res[2] = fork();
    int var = 0;

    if(res[0] == 0 && res[1] == 0 && res[2] == 0){
        var += 10;
    }
    else{
        var += 100;
    }

    printf("%d\n", var);
    return 0;
}
```

if(res[0]==0)

else if res[0]==0
 L fils
 fils pere
 else
 if res[1]==0
 fils
 else
 if res[2]==0
 fils
 else

2 pere → 1 fils → 2

1 + 2³ processus
→ 9

1. En exécutant ce programme on se rend compte que le père crée plus de 3 processus. En comptant le père, dites combien de processus sont créés par ce code et pourquoi.

910?

2. Donnez l'affichage de ce programme.

3. On a écrit le fichier **fils.c** suivant :

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char* argv[]){
    int var = atoi(argv[1]);
    printf("Le fils affiche : %d\n", var);
    return 0;
}
```

On voudrait remplacer le code des fils créés par **main.c** par le code de **fils.c**. Écrivez l'**exec1** correspondant. Vous devez fournir aux fils la variable **var**.

Vous n'êtes pas obligé de recopier tout le code. Donnez juste les lignes modifiées.

4. Donnez la (les) ligne(s) du terminal nécessaire pour compiler les fichiers **main.c** (avec le **exec1**) et **fils.c**.