

Le : 24/05/2014

Examen Rattrapage

Exercice 01 (5 points) (Questions de cours)

- 1 - Donner les objectifs du système d'exploitation?
- 2 - Quel est le rôle d'un PCB ? et quelles sont les trois principales catégories d'information d'un PCB ?
- 3 - Dans un système Ms-Dos, est ce qu'un processus qui fait une requête pour une ressource non disponible serait
- 4 - bloqué dans la file d'attente de cette ressource ? Expliquer.
- 5 - Quelles sont les états d'un processus ? *prêt, bloqué, active*
- 6 - Donner le rôle et l'objectif de l'MMU ? *active*

Exercice 02 (4 points) (l'édition de liens)

On dispose d'un ensemble de modules définis comme suit:

module PROGRAMME taille: 450 liens à satisfaire: OUVRI LIRE FERMER EDITER liens utilisables: RUN 12 ①	module ETIQUETTE taille: 957 liens utilisables: NOM 15 SOCIETE 223 ADRESSE 429 CODEPOST 912 VILLE 950 ④
module LECTURE taille: 460 liens utilisables: OUVRI 2 LIRE 250 FERMER 533 liens à satisfaire: NOM SOCIETE ADRESSE CODEPOST VILLE ②	module IMPRESSION taille: 214 liens utilisables: IMPRIMER 121 ⑤
module EDITION taille: 642 liens utilisables: EDITER 54 liens à satisfaire: NOM SOCIETE ADRESSE CODEPOST VILLE IMPRIMER ③	

- 1 - Les adresses d'implantations de ces modules après la construction de programme finale?
- 2 - La taille totale du programme résultant ? →
- 3 - La table des liens après la translation des modules?

taille
2723

Exercice 03 (5 points) (gestion de processus)

- Nous considérons un tableau qui contient les informations de quatre processus A,B,C,D :
- 1 - Dessiner (représenter) un schéma illustrant leur exécution à l'aide de :
 - l'algorithme FCFS.
 - l'algorithme SRT.
 - l'algorithme à tourniquet RR (quantum = 2).
 - 2 - Donner le temps de réponse moyen de ces processus dans chaque type d'ordonnancement.
 - 3 - Donner le temps d'attentes moyen de ces processus dans chaque type d'ordonnancement.

Processus	Date d'arrivée	Temps de traitement
A	0.000	4
B	2.001	7
C	3.001	2
D	3.002	2

Exercice 04 (6 points)

1. Comment transférer l'adresse logique vers l'adresse physique dans un système paginée ?
2. Donner le mécanisme du fonctionnement pour un système de segmentation paginée ?
3. Donner le nombre des processus, des instructions printf avec le schéma

```

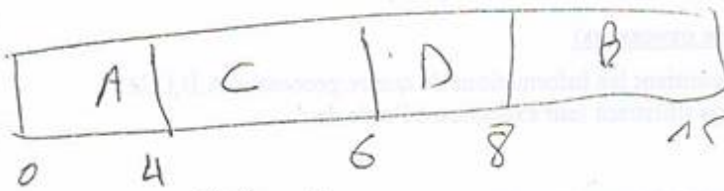
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>

int main() {
    int valeur;
    int valeur1;
    valeur1 = fork();
    printf("printf-0 \n");
    valeur = fork();
    printf("printf-0 \n");
    if (valeur == 0) {fork() ; printf("printf-0 \n");
                    fork() ; printf("printf-0 \n");}

    return 0;}

```

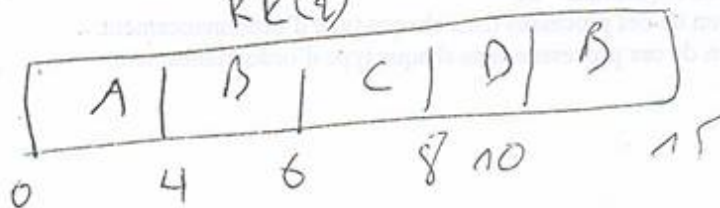
SAT



$TAM = 2,5$

$TRM = 6,25$

REC



$TAM = 3,5$

$TRM = 7,25$

Le : 17/06/2014

Examen Rattrapage

Exercice 01 (6 points) (Questions de cours)

1. Donner les objectifs du système d'exploitation?
2. Donner le rôle du vecteur d'interruption? Pourquoi les systèmes d'exploitation utilisant les niveaux d'interruption ?
3. Donner le rôle de l'ordonnanceur ?
4. Quelles sont les politiques de remplacement des pages victimes au niveau de mémoire virtuelle ?
5. Dans les deux cas où le $Q \ll (Q = \text{Quantum})$ et $Q \gg$ quelle est votre remarque au niveau d'ordonnanceur ?

Exercice 02 (4 points) (l'édition de liens)

On dispose de 5 modules compilés, mémorisés dans les fichiers `cmpdisk.o`, `cmpfile.o`, `disk_io.o`, `lsbrk.o`, `printstr.o`. Ces 5 modules constituent la base d'un programme, qui nécessite en plus des modules de bibliothèque.

On se propose de faire l'édition des liens des 5 modules dans l'ordre suivant :

`cmpdisk.o`, `cmpfile.o`, `disk_io.o`, `lsbrk.o`, `printstr.o`.

- 1- Définir les adresses d'implantations des modules après la construction de programme final ?
- 2- Donner le contenu de la table des liens après la construction de programme final ?
- 3- la taille total de programme

<code>cmpdisk.o:</code>	<code>las</code>	<code>bloc_transfer_dr</code>	
	<code>las</code>	<code>close_printer</code>	
	<code>las</code>	<code>compare_file</code>	
	<code>lu</code>	<code>compare_hierarch</code>	1062
	<code>las</code>	<code>exit</code>	
	<code>lu</code>	<code>exit_prog</code>	2020
	<code>lu</code>	<code>file_error</code>	422
	<code>las</code>	<code>free</code>	
	<code>lac</code>	<code>get_memory</code>	
	<code>lu</code>	<code>main</code>	2460
	<code>las</code>	<code>open_drive</code>	
	<code>las</code>	<code>open_printer</code>	
	<code>las</code>	<code>print_string</code>	
<code>cmpfile.o:</code>	<code>las</code>	<code>bloc_transfer_dr</code>	
	<code>lu</code>	<code>compare_file</code>	200
	<code>las</code>	<code>file_error</code>	
<code>disk_io.o:</code>	<code>lu</code>	<code>bloc_transfer_dr</code>	3118
	<code>las</code>	<code>exit</code>	
	<code>las</code>	<code>exit_prog</code>	
	<code>las</code>	<code>get_memory</code>	
	<code>lu</code>	<code>open_drive</code>	236
<code>lsbrk.o:</code>	<code>las</code>	<code>print_string</code>	
	<code>las</code>	<code>exit_prog</code>	
	<code>lu</code>	<code>get_memory</code>	0
	<code>las</code>	<code>malloc</code>	
<code>printstr.o:</code>	<code>las</code>	<code>print_string</code>	
	<code>lu</code>	<code>ask_confirm</code>	750
	<code>lu</code>	<code>close_printer</code>	1020
	<code>las</code>	<code>getchar</code>	
	<code>lu</code>	<code>open_printer</code>	996
	<code>lu</code>	<code>print_string</code>	340
<code>las</code>	<code>write</code>		

<code>cmpdisk.o</code>	3376
<code>cmpfile.o</code>	644
<code>disk_io.o</code>	3862
<code>lsbrk.o</code>	82
<code>printstr.o</code>	1196

table 2. résultat de "taille"

Exercice 03 (5 points) (gestion de processus)

Soit un algorithme d'ordonnement préemptif basé sur des priorités dynamiques, où une plus grande valeur indique une priorité plus grande. Quand un processus entre dans la file des processus prêts, sa priorité est 0. Tant qu'il reste dans cette file, sa priorité augmente graduellement en fonction du temps avec un taux α . Lorsqu'il passe à l'état d'exécution, sa priorité continue d'augmenter, mais avec un taux différent β .

1. Indiquez quel est l'algorithme d'ordonnement obtenu si $0 < \alpha < \beta$. S'il s'agit d'un algorithme équivalent à un algorithme déjà vu en cours. Indiquez-le.
2. Supposez maintenant que $\alpha > 0$ et que la valeur de priorité est une valeur constante $P > 0$ lorsqu'un processus passe à l'état d'exécution. Répondez aux questions suivantes :
 2. Quel algorithme obtenons-nous dans ce cas ?
 3. Décrivez un avantage et un problème important de cette approche.

Exercice 04 (5 points)

1. Donner le nombre des processus avec le schéma.
2. Donner le nombre des instructions printf avec le schéma.

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

int main(void) {
    printf("pere \n");
    int valeur;
    int valeur1;
    valeur = fork();
    if (valeur == 0) { printf("p=0 \n"); }
    if (valeur > 0) { printf("p>0 \n"); fork(); }
}

valeur1 = fork();
valeur1 = fork();

if (valeur1 == 0) {
    int valeur1 = fork (); printf("p1=0 \n"); }
if (valeur1 > 0) { printf("p1>0 \n"); fork(); }

return 0;}
```

Bon courage

P0
P1
P2
P3
P4
P5
P6
P7
P8
P9
P10
P11
P12
P13
P14
P15
P16
P17
P18
P19
P20