Université de M'sila
Faculté : MI Département : d'informatique
Année Universitaire : 2023 / 2024
2ème Année Licence (2L)

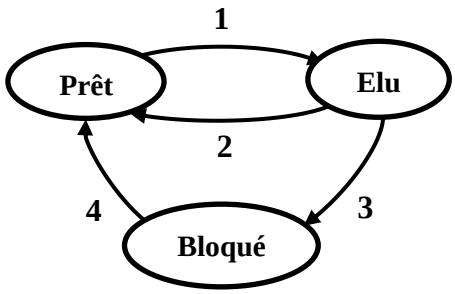
Nom : ……
Prénom :……...……
Groupe : ……..……...

Examen de Systèmes d'Exploitation 1

Date : 21/05/2024 **Durée: 1h30 - Documentation non autorisée**

Exercice 1 : (Questions de Compréhension : 5 pts) (10 minutes)

Q1) Etant donné le diagramme d'états/transitions suivant, citer la/les transition(s) qui doivent être supprimée(s) si on utilise un algorithme d'ordonnancement sans réquisition (non préemptif). Justifier votre réponse. **(1 pt)**



La transition à supprimer est **la transition (2)** qui passe de l'état **Elu à l'état Prêt**, car cette transition a uniquement lieu lors d'une préemption.

Q2) Quels sont les avantages et inconvénients du choix d'un quantum petit pour l'algorithme de scheduling Round Robin ? **(1 pt)**

Avantage : partage du processeur (Équite entre les processus) : Chacun des utilisateurs a l'impression de disposer de son propre processeur.

Inconvénients : surcharge du système du aux fréquentes commutations de contexte.

Q3) Dans quel cas est-il intéressant de masquer une interruption ? **(1 pt)**

Par exemple, pour éviter que la routine d'interruption elle-même ne soit interrompue par une autre interruption (qu'il faut masquer).

Q4) Question : Qu'est-ce qu'un déroutement ? **(1 pt)**

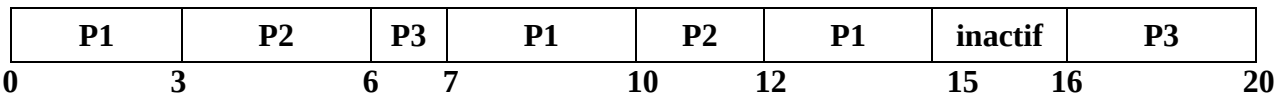
Un déroutement est un type d'interruption interne (provoquée par le processus lui-même). Il nécessite l'intervention du système d'exploitation.

Q5) Décrivez ce qui se passe, du côté du système d'exploitation, lorsqu'une touche de clavier est pressée : **(1 pt)**

Après chaque touche pressée, une interruption (de type matérielle associée au clavier) est générée. Le processeur interrompt son traitement pour lancer la routine d'interruption associée.

Exercice 2 : (Ordonnancement : 5 pts) (25 minutes)

La figure suivante représente le diagramme de Gantt d'un scheduling du processeur utilisant l'algorithme « **Round Robin** » et trois processus : **P1**, **P2** et **P3**. (Si les processus arrivés en même temps ont file d'attente, le système prend l'ordre d'arrivée des processus P1, P2, P3)



Q1) Quelle est la durée du quantum ? **(0.25 pts)**

03 [UT]

Q2) Quel est le temps d'attente du processus **P1** ? **(0.25 pts)**

06 [UT]

Q3) Quel est le temps de réponse du processus **P2** ? Justifiez. **(0.75 pts)**

Le processus P2 peut être arrivé à l'instant 0, 1 ou 2. Le temps de réponse en dépend : il peut être 12, 11 ou 10 [UT].

Q4) Quel est le temps d'attente du processus **P3** ? Justifiez. **(0.75 pts)**

Le processus P3 peut être arrivé à l'instant 0, 1 ou 2. Le temps d'attente en dépend : il peut être 6, 5 ou 4 [UT].

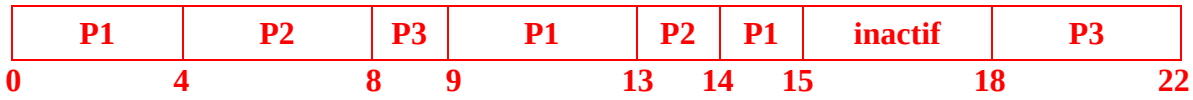
Q5) Que s'est-il passé entre les instants **t = 15** et **t = 16** ? Justifiez. **(1 pt)**

A l'instant t = 15, les processus P1 et P2 sont terminés ; le seul processus restant est le processus P3, mais il est en attente d'une opération d'entrée/sortie ou d'un évènement. Le processeur est donc inactif jusqu'à la reprise de P3 à l'instant t = 16.

Q6) Quel est l'état du processus **P3** à l'instant **t = 9**. **(0.5 pts)**

A l'instant t = 9, le processus P3 est en attente d'une opération d'entrée/sortie ou d'un évènement

Q7) Dessinez le diagramme de Gantt du même problème, mais en considérant un quantum égal à **4**. **(1.5 pts)**



Exercice 3 : (La primitive fork : 3 pts) (20 minutes)

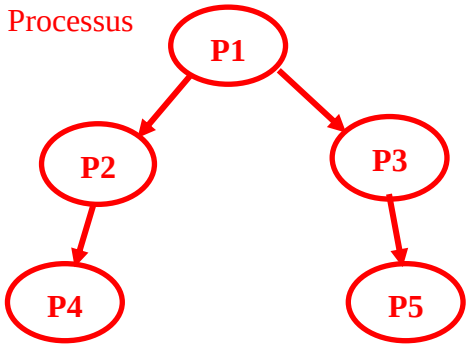
Dans cet exercice on suppose que les numéros des **PIDs** attribués aux processus sont strictement croissants. Si un père a un **PID (n)**, son fils aura un **PID (m)** tel que (**m > n**). Le premier processus fils sera créé aura un PID inférieur au seconde processus fils créé (PID(Père) = **n** ⇒ PID(Fils1) = **n+1**, PID(Fils2) = **n+2**, ...etc.).

Soit le code suivant :

```
int main ()
{
   pid_t p1 , p2 ;
   p1 =  fork ( ) ;
   p2 = fork ( ) ;
   if ( ( p1- p2 ) > 0 )
      fork ( ) ;
    printf ("Je suis %d : p1 = %d , p2 = %d \n",getpid( ), p1, p2) ;
   return 0 ;
}
```

Q1) Donnez le nombre ainsi que l'arborescence des processus créés par ce code. **(1.5 pts)**

5 Processus



Q2) Proposez pour chaque processus créé son affichage à l'écran. Nous rappelons que les **PIDs** sont affichés comme suit : *PID-P1=**101**, PID-P2=**102**, PID-P3=**103**, PID-P4=**104**, ...etc.*

Par exemple, le processus P8 (PID-P8=**108**) avec p1 (PID-P5) = **105** et p2=**0** aura pour affichage :

Je suis 108 : p1 = 105, p2 = 0. **(1.5 pts)**

Je suis 101, p1 = 102, p2 = 103.

Je suis 102, p1 = 0, p2 = 104.

Je suis 103, p1 = 102, p2 = 0.

Je suis 104, p1 = 0, p2 = 0.

Je suis 105, p1 = 102, p2 = 0.

Exercice 4 : (Algorithmes de Remplacement de Pages : 7 pts) (30 minutes)

On considère une mémoire paginée possédant **N** cadres de pages. Soit un programme possédant un espace virtuel de **768 Octets** et la taille de page est de **64 Octets**. Le programme fait référence, durant son exécution, aux adresses virtuelles suivantes :

60, 196, 90, 450, 220, 490, 330, 390, 30, 345, 230, 440, 360, 10

Q1) Donnez le nombre de pages de ce programme ? **(0.25 pts)**

Nombre de pages = Taille MV / Taille de page = 768/64 = **12 pages.**

Q2) Donnez la suite des numéros de pages référencés ? **(1 pt)**

Donc, il suffit de diviser l'adresse par 64, ce qui donne la suite des numéros de pages référencés suivante :

0, 3, 1, 7, 3, 7, 5, 6, 0, 5, 3, 6, 5, 0

Q3) Combien de défauts de pages peuvent se produire au minimum ? Justifiez. **(0.5 pts)**

Au minimum, on a **6 défauts de pages**.

Justification : Chacune des 6 pages constituant la chaine de références provoquera certainement un défaut de pages lors de la première utilisation.

Q4) Combien de défauts de pages peuvent se produire au maximum ? Justifiez. **(0.5 pts)**

Au maximum, on a **14 défauts** de pages.

Justification : Cela arrivera lorsque N=1 (un seul cadre de page). Chaque page de la chaine provoquera ainsi un défaut de page.

Sachant que la taille de la mémoire physique est de **256 Octets.**

Q5) Donnez le nombre de cadres (**N**) ? **(0.25 pts)**

Nombre de cadres (N) = Taille MPhy / Taille de page =256/64 = **4 cadres**

Q6) En prenant **N=3**, déterminez le nombre de défauts de page génères en applique les algorithmes de remplacement **FIFO**, **LRU,** et **FIFO de la seconde chance**. **(3 pts)**

FIFO : le nombre de défauts de page = 9 (1 pt)														
Références	0	3	1	7	3	7	5	6	0	5	3	6	5	0
Cadre 1	0	0	0	7	7	7	7	7	0	0	0	0	0	0
Cadre 2		3	3	3	3	3	5	5	5	5	3	3	3	3
Cadre 3			1	1	1	1	1	6	6	6	6	6	5	5
Défaut de page	D	D	D	D			D	D	D		D		D	

LRU : le nombre de défauts de page = 10 (1 pt)														
Références	0	3	1	7	3	7	5	6	0	5	3	6	5	0
Cadre 1	0	0	0	7	7	7	7	7	0	0	0	6	6	6
Cadre 2		3	3	3	3	3	3	6	6	6	3	3	3	0
Cadre 3			1	1	1	1	5	5	5	5	5	5	5	5
Défaut de page	D	D	D	D			D	D	D		D	D		D

FIFO de la seconde chance : le nombre de défauts de page = 11 (1 pt)														
Références	0	3	1	7	3	7	5	6	0	5	3	6	5	0
Cadre 1	0_1^+	0_1^+	0_1^+	7_1	7_1	7_1	7_1^+	7_0	0_1	0_1	0_0	0_0^+	5_1	5_0
Cadre 2		3_1	3_1	3_0^+	3_1^+	3_1^+	3_0	6_1	6_1^+	6_1^+	3_1	3_1	3_1^+	0_1
Cadre 3			1_1	1_0	1_0	1_0	5_1	5_1^+	5_0	5_1	5_0^+	6_1	6_1	6_0^+
Défaut de page	D	D	D	D			D	D	D		D	D	D	D

Q7) On suppose que le temps de chargement d'une page du disque vers la mémoire est **X** et que le temps de sauvegarde d'une page de la mémoire vers le disque est **Y**. Calculer le temps total de traitement (en fonction de **X** et **Y**) de la chaine de références pour l' algorithme **LRU**. **(1.5 pts)**

LRU	**Références**	0	3	1	7	3	7	5	6	0	5	3	6	5	0
	Défaut de page	D	D	D	D			D	D	D		D	D		D
	Temps de traitement	X	X	X	X + Y			X + Y	X + Y	X + Y		X + Y	X + Y		X + Y

Temps Total de Traitement = 10 X + 7 Y [U.T]

Exercice 5 : (Bonus : 2 pts) (5 minutes)

Si un système utiliser l'algorithme du travail le plus court d'abord pour l'ordonnancement à court terme et une moyenne exponentielle de $\alpha = 0.5$, quel est le prochaine cycle processeur estimé pour un processus aux cycles processeurs **5**, **8**, **3** et **5** et une valeur initiale de **10** pour e_1. **(2 pts)**

Les différentes valeurs de e sont les suivants :

$e_1 = 10 \quad ==> \quad e_2 = 0.5 \times 5 + 0.5 \times 10 = 7.5$

$e_2 = 7.5 \quad ==> \quad e_3 = 0.5 \times 8 + 0.5 \times 7.5 = 7.75$

$e_3 = 7.75 \quad ==> \quad e_4 = 0.5 \times 3 + 0.5 \times 7.75 = 5.375$

$e_4 = 5.375 \quad ==> \quad e_5 = 0.5 \times 5 + 0.5 \times 5.375 = 5.1875$

Bon courage