Réponses aux questions

Partie I

Exercice 1 Threads

a) Le pourcentage d'exécution de P1 est 2/3 (66%) puisqu'il a deux threads et les threads sont des threads système

b) Le pourcentage d'exécution de P1 est 1/2 (50%) puisqu'il a deux threads et les threads sont des threads utilisateur

c) La sortie ressemblera à:

aXbYcXdYeXfYgXhYiX

ou comme

aYbXcYdXeYfXgYhXiY

en fonction du thread qui va le premier. Notez que le dernier caractère (j) n'apparaîtra pas sur la sortie car le premier thread qu'exécute Exit terminera le processus (les deux threads).

d)

Exercice 2 Segmentation paginée

1. Le numero du segment

   8212 = 2*4096 + 20. Le segment S1

2. Le numéro de page dans le segment

   La page 3

3. Le déplacement dans la page
   20

4. Le numéro de case
   0

5. Le déplacement dans la case
   20

6. L'adresse physique (en décimal et en binaire)

   0000 0000 0001 0100

sur 16 bit puisque la taille de la mémoire physique est 64 KO

Exercice 3 Synchronisation

a) Les actions Ai ne soient jamais simultanées (0.5 pt)

Init(mutex,1)
Pi P(mutex)
Ai
V(mutex)

b) Les actions Ai ne soient jamais simultanées et se déroulent toujours dans l'ordre A1A2A3A1A2A3... (1 pt)

Init(S1,1) Init(S2,0) Init(S3,0)
P1 : P(S1) ; A1 ; V(S2)
P2 : P(S2) ; A2 ; V(S3)
P3 : P(S3) ; A3 ; V(S1)

c) Les actions Ai ne soient jamais simultanées et se déroulent toujours dans l'ordre A1(A2 ou A3)A1(A2 ou A3)... (1,5 pt)

Init(S1,1) Init(S,0)
P1 : P(S1) ; A1 ; V(S)
P2 : P(S) ; A2 ; V(S1)
P3 : P(S) ; A3 ; V(S1)

d) Les actions Ai se déroulent toujours séquentiellement mais dans un ordre quelconque, par exemple : (A2A1A3)(A2A3A1)... (2 pt)

Init(mutex, 1) Init(S1,2) Init(S2,2) Init(S3,2)
P1 : P(S1) ; P(S1) ; P(mutex) ; A1 ; V(mutex) ; V(S2) ; V(S3) ;
P2 : P(S2) ; P(S2) ; P(mutex) ; A2 ; V(mutex) ; V(S1) ; V(S3) ;
P3 : P(S3) ; P(S3); P(mutex) ; A3 ; V(mutex) ; V(S1) ; V(S2) ;

Partie II

Exercice 1 Ordonnancement deprocessus

1. C'est la moyenne des temps de traitement d'un ensemble de processus. Le temps de traitement est la durée passée entre l'arrivée d'un processus et la fin de son exécution.

2. Si l'on est dans un système batch par exemple.

3.

File	Processus[temps arrivée][Durée]	Algorithme s'appliquant dans la file
F3	P2[0] [4] P5[0.5] [2] P8[8.5] [4]	SRT Quantum=1
F2	P1[0] [7] P4[1] [1] P7[2] [1]	RR Quantum=2

F1	P3[1] [6] P6[2] [4]	FCFS

Diagramme d'exécution

Temps	0	1	2	3	4	5	6	7	8	9	10	11
Processus	P2	P5	P5	P2	P2	P2	P1	P1	P4	P8	P8	P8

12	13	14	15	16	17	18	19	20	21	22	23	24
P8	P7	P1	P1	P1	P1	P1	P3	P3	P3	P3	P3	P3

25	26	27	28	29
P6	P6	P6	P6	FIN

TTM= [(19-0) + (6-0) + (25-1) + (9-1) + (3-0.5) + (29-2) + (14-2) + (13-8,5)]/8 =103/8 = 12,875

Exercice 2 Inter-blocage

a) i) Tous les processus sont bloqués en mémoire centrale, aucun processus ne peut être retiré ou chargé faute de place.

ii) Un processus demande la création d'un processus fils, il n'y a pas assez d'espace en mémoire swap et il n'y a pas assez d'espace en mémoire centrale pour charger les processus prêts de la mémoire swap.

b) i) Lorsqu'un processus est chargé en mémoire centrale, l'espace qu'il occupait en mémoire swap n'est pas libéré.

ii) Toute demande de création de processus est refusée s'il n'y a pas assez d'espace en mémoire swap.

1. a) Déroulement de l'algorithme du Banquier de détermination d'un état sain :

TRAVAIL := DISPONIBLE = (1, 5, 2, 0) ;
FINI := (F, F, F, F, F) ;
i :=0 ; BESOIN[0]:= MAX[0]-ALLOCATION[0] = (0, 0, 0, 0) <= TRAVAIL
TRAVAIL := TRAVAIL + ALLOCATION[0] = (1, 5, 3, 2) ;
FINI := (V, F, F, F, F);
i:=2; BESOIN[2]:= MAX[2]-ALLOCATION[2] = (1, 0, 0, 2) <= TRAVAIL
TRAVAIL := TRAVAIL + ALLOCATION[2] = (2, 8, 8, 6) ;
FINI := (V, F, V, F, F) ;
i :=1 ; BESOIN[1]:= MAX[1]-ALLOCATION[1] = (0, 7, 5, 0) <= TRAVAIL
TRAVAIL := TRAVAIL + ALLOCATION[1] = (3, 8, 8, 6) ;
FINI := (V, V, V, F, F) ;
i :=3; BESOIN[3]:= MAX[3]-ALLOCATION[3] = (0, 0, 2, 0) <= TRAVAIL
TRAVAIL := TRAVAIL + ALLOCATION[3] = (3, 14, 11, 8) ;

FINI := (V, V, V, V, F) ;
i:=4 ; BESOIN[4]:= MAX[4]-ALLOCATION[4] = (1, 6, 4, 2) <= TRAVAIL
TRAVAIL := TRAVAIL + ALLOCATION[4] = (3, 14, 12, 12) ;
FINI := (V, V, V, V, V) ;

L'état est sain et une séquence saine est : P0, P2, P1, P3, P4

2. b) Déroulement de l'algorithme du Banquier, Algorithme de requête

DEMANDE1 = (0,5,2,0) <= BESOIN[1]= MAX[1]-ALLOCATION[1] = (0,7,5,0)
DEMANDE1 <= DISPONIBLE = (1, 5, 2, 0)
DISPONIBLE := DISPONIBLE - DEMANDE1 = (1, 0, 0, 0) ;
ALLOCATION[1] := ALLOCATION[1] + DEMANDE1 = (1,5,2,0) ;
BESOIN[1] := BESOIN[1] - DEMANDE1 = (0, 2, 3, 0) ;
Vérification si le nouvel état est sain
TRAVAIL := DISPONIBLE = (1, 0, 0, 0) ;
FINI := (F, F, F, F, F) ;
i :=0 ; BESOIN[0]:= MAX[0]-ALLOCATION[0] = (0, 0, 0, 0) <= TRAVAIL
TRAVAIL := TRAVAIL + ALLOCATION[0] = (1, 0, 1, 2) ;
FINI := (V, F, F, F, F);
i:=2; BESOIN[2]:= MAX[2]-ALLOCATION[2] = (1, 0, 0, 2) <= TRAVAIL
TRAVAIL := TRAVAIL + ALLOCATION[2] = (2, 3, 6, 6) ;
FINI := (V, F, V, F, F) ;
i :=1 ; BESOIN[1] <= TRAVAIL
TRAVAIL := TRAVAIL + ALLOCATION[1] = (3, 8, 8, 6) ;
FINI := (V, V, V, F, F);
i :=3; BESOIN[3]:= MAX[3]-ALLOCATION[3] = (0, 0, 2, 0) <= TRAVAIL
TRAVAIL := TRAVAIL + ALLOCATION[3] = (3,14,11,8) ;
FINI := (V, V, V, V, F) ;
i:=4 ; BESOIN[4]:= MAX[4]-ALLOCATION[4] = (1, 6, 4, 2) <= TRAVAIL
TRAVAIL := TRAVAIL + ALLOCATION[4] = (3, 14, 12, 12) ;
FINI := (V, V, V, V, V) ;

Le nouvel état est sain alors allocation avec succès.