Faculté des Sciences ,
Département d'Informatique

L3 ISIL
2023/2024

Nom et prénom :

Groupe :

Epreuve de moyenne durée en Systèmes d'exploitation 2 (1h30 minutes)

Questions de cours (4pts)

a- Parmi les mécanismes et les notions vus en cours lesquels répondent à ces définitions **(0.5pt*6)**

1- Si aucun processus ne détient le verrou, et qu'un processus demande à entrer dans sa section critique, alors ce processus doit éventuellement pouvoir entrer dans sa section critique

Réponse : Verrou mous

2- Met en attente passive les processus dans le moniteur ou bien les réveille
Réponse : condition d'attente

3- L'idée de base de cet algorithme est que chaque processus a une variable flag pour indiquer s'il souhaite entrer dans sa section critique. De plus, il y a une variable turn qui détermine quel processus a la priorité pour entrer dans sa section critique
Réponse : peterson

4- Situation qui peut se produire lorsqu'un processus ou un thread est constamment [illegible] ou retardé, même s'il est prêt à avancer.
Réponse : interblocage

5- Mécanisme de communication inter-processus (IPC) permettant à deux processus de communiquer en se transmettant des données de manière unidirectionnelle
Réponse : canal unidirectionnelle ou pipe

6- Fonction qui permet de lier le socket à un point de communication local défini par une adresse et un port
Réponse : Bind()

b- Dans quelles situations se trouvent ces deux codes si les read sont bloquants? **(0.5pt*2)**

Code 1	**Code 2**

int p[2];
pipe (p) ; /* le tube est vide est le processus en est le seul écrivain */
read (p[0], buflecture, 1);
write (p[1], bufecriture, 1);
	int tube1[2];int tube2[2];
	pipe(tube1);pipe(tube2);
	if (fork() == 0)
	{
	read (tube2 [0],buf3,1); // tube 2 vide
	write(tube2[1],buf4,1) ;
	}
	else
	{
	read(tube1[0],buf1,1); // tube 1 vide
	write(tube2[1],buf2,1);
	}

Situation : enfant est bloqué	**Situation :** pere est bloqué

Exercice 1 : Multiprocessing/Multithreading (2 +2.5 pts)

1- Soit le code suivant , donner l'arborescence des fils généré en montrant ce que chacun affiche

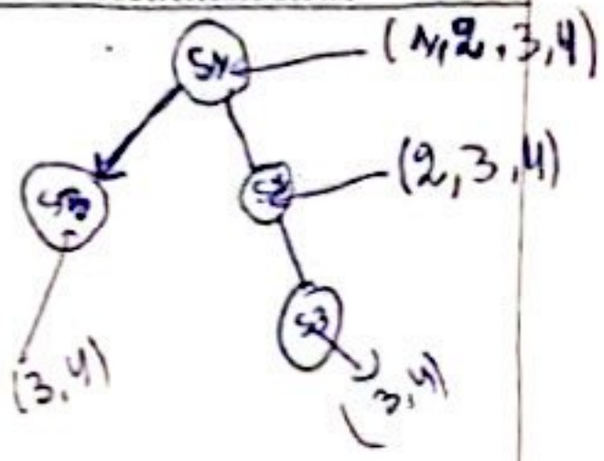
Programme	Arborescence des fils

```
int main()
{
    if (!fork())
    {
        printf("1 ");

        if (!fork() || fork())
        {
            printf("2 ");
            fork();
        }

        printf("3");
    }

    printf("4 ");
    return 0;
}
```



2- Le nombre d'arrangements d'un ensemble E comprenant n éléments pris k à la fois est donné par la formule : $A_n^k = \frac{n!}{(n-k)!}$. Pour accélérer les calculs , on propose d'utiliser deux threads : un pour calculer n ! et l'autre pour calculer (n-k) !

Fonction Factoriel	Fonction NbArrangements

```
// Fonction pour calculer le factoriel
Function factoriel(* n_ptr args) {
  int n = * n_ptr ;
  int i,fact;
  int *res = malloc(sizeof(int));
  fact=1;

  for (i = 1; i <= n; i++) {
    fact = fact * i;
  }
  *res=fact;
  return *res ;
}
```

```
int nbArrangements(int n, int k)
{
  pthread_t t1, t2;
  int *n_ptr = malloc(sizeof(int));
  int *k_ptr = malloc(sizeof(int));
  *n_ptr = n;
  pthread_create( t1 , n_ptr , null , null );
  *k_ptr = n-k;
  pthread_create( t2 , k_ptr , null , null );

  int *n_fact, *k_fact;
  int result;
  pthread_join (t1, *n_fact, null);
  pthread_join (t2, *k_fact, null);
  result = *n_fact / (*k_fact );
  return result;
}
```

Exercice 2 : Synchronisation (8.5 pts)

Un boulanger desire organiser les arrivées de ses clients. On souhaite réguler le service des clients avec des sémaphores.

1- Quand il n'y a pas de monde durant la journée, le boulanger fait passer les femmes et les hommes dans une seule file d'attente. Ils seront servis selon leur ordre d'arrivée

a- Un sémaphore seulement(**mutex**) est nécessaire , Pourquoi ? Donner sa valeur initiale(0.5pt)
Réponse : Pour Say if the next element can enter or no , valeur initiale 0

b- Compléter le code nécessaire pour simuler le comportement d'un client. **(1 pt)**

Processus client
// Arrivée du client P(SF) P(Sm) V(SF) // fin de l'attente Boulanger.Servir() // départ du client V(SH)

2- En fin de journée, le boulanger fait passer les femmes en priorité. Il va séparer les hommes des femmes en deux files d'attente distinctes. Une pour les hommes et la seconde pour les femmes. S'il n'y a aucune femme, il fera passer les hommes (selon leur ordre d'arrivée). On propose d'utiliser **3 sémaphores** et une variable nbFemmes pour compter le nombre de femmes.

a- Complétez le code nécessaire pour les clients hommes et femmes et donner la valeur initiale des sémaphores utilisés

Sémaphores : SM1=1 SM2=0 SM3=N NbrP=0 (0.5 pt)

Processus client_homme(1.5pts)	Processus client_femme(2.5pts)
// Arrivée du client P(SM2) P(SM1) V(SM1) // fin de l'attente Boulanger.Servir() // départ du client V(SM2)	// Arrivée de la client P(SM1) NbrP++ P(SM2) P(SM3) V(SM1) // fin de l'attente Boulanger.Servir() // départ de la cliente NbrF-- while (NbrP)0); V(SM3) V(SM1)

b- Quelle situation peut provoquer un tel scénario ? algorithme de Boulangé ..... **(0.5 pt)**

3- Des clientes se donnent régulièrement un rdv pour aller à la boulangerie ensemble. La cliente B doit attendre la cliente A pour sortir ensemble. La cliente D doit attendre l'arrivée de la cliente C et la cliente B pour sortir de chez elle.

a- Donner le graphe d'attente associé au problème **(0.5 pt)**

b- Proposer une solution en utilisant un nombre minimum de sémaphore **(1.5 pt)**

Sémaphores : SA=0 SB=0 SC=0

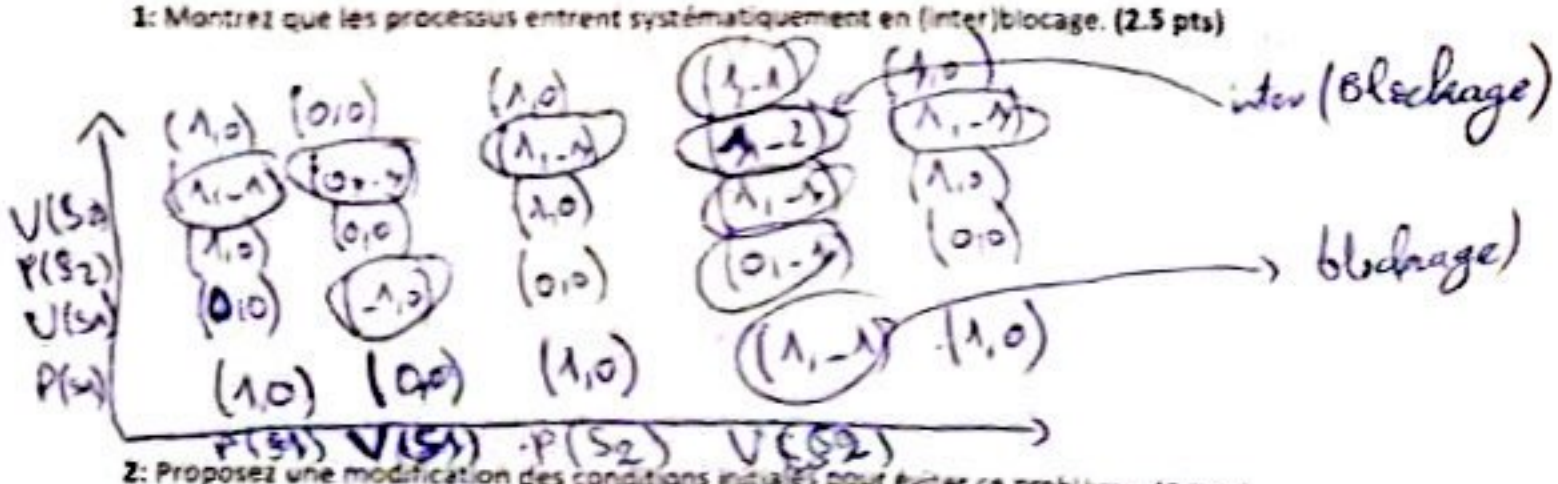
Cliente A	Cliente B	Cliente C	Cliente D
V(SA)	P(SA) V(SB)	V(SC)	P(SB) P(SC)

Exercice 3 : Interblocage (3 pts)

Considérez le programme suivant, qui utilise deux sémaphores pour obtenir une exclusion mutuelle.

Valeur initiale des sémaphores	Processus 1	Processus 2
S1=1 S2=0	P(S1) V(S1) P(S2) V(S2)	P(S1) V(S1) P(S2) V(S2)

1: Montrez que les processus entrent systématiquement en (inter)blocage. **(2.5 pts)**



2: Proposez une modification des conditions initiales pour éviter ce problème. **(0.5 pt)**

(S1, S2) = (1,0) ⟹ (1,1) car le problème est dans S2