

Examen final de Programmation Système

Informatique 2^{ème} année 2019/2020

—Mathieu Faverge - mfaverge@enseirb-matmeca.fr—

Le présent examen peut être réalisé avec l'aide des documents de cours et de TDs et bien entendu doit être réalisé sans l'aide de son voisin ! La durée de l'examen est de 2h.

Les réponses aux questions doivent se faire sur le sujet d'examen à l'intérieur des cadres prévus pour cela. N'oubliez pas pour cela de remplir l'entête de chacune des feuilles. Si vous n'avez pas assez de place, c'est probablement que vous en avez trop mis.

Le barème (sur 41 pts) est approximatif et n'est là que pour donner un ordre d'idée par exercice. Une fonction le ramenant sur 20 sera appliquée et n'est pas encore définie.

L'ensemble des fonctions à utiliser dans les codes à fournir est présent dans vos slides de cours. Seul les paramètres important seront pris en considération, les autres pourront être omis. On pourra également utiliser la fonction `exit_if()` pour tester les retours des appels systèmes.

Lire l'intégralité du sujet avant de commencer. Il n'est pas nécessaire de répondre dans l'ordre même si cela est recommandé.

Nom :

Prénom :

Groupe :

►Exercice 1. Questions de cours (Échauffement) (15pts)

Répondez aux questions suivantes de manière claire et concise en respectant l'espace donné pour la réponse.

1. Classez les routines suivantes entre celles qui sont des appels systèmes et celles qui sont des fonctions utilisateurs: `printf`, `write`, `recv`, `open`, `fclose`, `shmat`, `fork`, `fprintf`, `exit`, `_exit`. (2 pts)

Appels systèmes:

Appels utilisateurs:

2. Rappelez quelles sont les différentes sections qui composent l'espace d'adressage virtuel d'un processus et leur fonction respective. Dans quelle section se trouve la mémoire allouée par l'appel à `malloc` (2 pts).

3. Pourquoi est-il fortement déconseillé d'utiliser les fonctions `signal()` et `pause()` ? Quelles fonctions faudrait-il utiliser à la place ? (2pts)

4. Quels sont les affichages possibles de ce programme ? Justifiez chacun de ces affichages. (2pts)

5. Que se passe-t-il si on inverse les instructions `alarm(aleatoire(3))` et `fibonacci(aleatoire(42))` ? Justifiez. (2pts)

► **Exercice 3.** Threads et concurrence - Boucherie (18pts)

On souhaite simuler le fonctionnement d'une boucherie en modélisant le comportement des clients au moyen de threads: leur nombre est aléatoire, tout comme le moment où chacun d'eux se décide à se rendre à la boucherie. Chaque thread exécute la fonction `client` (décrite ci-après) puis se termine.

```
1 int nbclients = 0;
2 int boucher_libre = 1;
3
4 void client ()
5 {
6     if ( nbclients == MAX_CLIENTS )
7         return; // trop de monde, on reviendra plus tard
8
9     nbclients++;
10
11     while( boucher_libre == 0 ) {
12         /* on prend place dans la file d'attente */;
13     }
14
15     boucher_libre = 0;
16     sleep( SE_FAIRESERVIR );
17     boucher_libre = 1;
18     nbclients--;
19 }
```

Nom:

Prénom:

Groupe:

1. L'exécution de ce code avec plusieurs clients ne fonctionne pas correctement. Quel est le problème ? Et que faudrait il faire (sans nécessairement donner le code) pour corriger ce problème ? (2pt)

2. La boucherie ayant beaucoup de succès, le boucher a engagé plusieurs apprentis pour l'aider à répondre aux besoins des clients. On considère donc désormais qu'on a NB_BOUCHERS disponibles pour servir les clients. Donnez une implémentation de la fonction `client()` qui soit correcte et qui permette de réduire le temps moyen d'attente des clients. (6pts)

3. On souhaite désormais que les clients ne se doublent pas dans la file d'attente, c'est-à-dire qu'ils se fassent servir dans l'ordre de leur arrivée. Une idée est d'utiliser des tickets numérotés qui permettent à chaque client de récupérer un numéro unique, ainsi que d'utiliser un afficheur indiquant le numéro du prochain client autorisé à se faire servir. Donnez la nouvelle version du code. Expliquez le rôle de chaque variable introduite. (6pts)

Questions bonus pour ceux qui ont écouté en amphi.

4. Proposez une modification pour éviter l'attente active de la ligne 12. (2pts)

5. On souhaite désormais utiliser un processus pour la boucherie, et un processus pour chaque client. Expliquez les mécanismes qu'il faudrait mettre en oeuvre pour adapter la fonction `client()` à une utilisation entre processus. (2pts)