CTP PRS3

Déroulement du CTP

1. Après avoir démarré votre ordinateur sous `Ubuntu`.
2. Et après avoir **téléchargé le ZIP du CTP** et vérifié qu'il contient les éléments suivants :
   a. Le **sujet** du CTP et le fichier **`coiffeur.c`.**
   b. Le **support du cours en PDF** (**c'est le seul document autorisé avec l'aide en ligne**).
3. **Débranchez** physiquement le **câble réseau** de votre machine (**aucune communication — quelque soit sa forme — n'est autorisée**).

4. Vous êtes prêt à composer, en respectant les règles qui suivent :
   a. **Créez** un **répertoire** de travail avec un nom de la forme `CTP_LSE_MON_NOM`
   b. **Créez** un fichier pour y rédiger **vos réponses**. **Le fichier à rendre doit être au format PDF !**
   c. **Commencez par lire le sujet en entier pour en avoir une vue d'ensemble**.
5. Composez.

6. A la fin de la composition, **compressez** votre répertoire contenant le fichier de **réponses** portant votre **NOM**, ainsi que les fichiers source **`C`** et tout autre fichier utile.
7. **Branchez** le **câble réseau** de votre machine
8. **Déposez** votre travail sur la plateforme moodle, en utilisant le lien prévu à cet effet.
   Attention à l'heure de dépôt de votre travail, le retard de remise sera sanctionné

Bon travail à tous
Thomas, vous dirait : « Enjoy ! »

Problème du coiffeur endormi (Processus / Sémaphores / Signaux)

Soit, un salon de coiffure, qui est équipé d'un **seul** fauteuil **-F-** (client coiffé) et **N** chaises (clients en attente).

(1) Quand il n'y a pas de clients, le coiffeur **-C-** s'endort dans le fauteuil (chose qu'il commence par faire dès son arrivée au matin) ;
(2) Le premier client doit réveiller le coiffeur à son arrivée. S'il y a des clients et qu'il y a une chaise vide alors il attend son tour (éveillé, le client);
(3) Si le salon est plein (pas de chaise d'attente vide), le client repart sans se faire coiffer ;
(4) Quand le dernier client est servi, le coiffeur s'endort de nouveau dans le fauteuil.

Vous allez construire une solution à ce problème de manière progressive.

Question n°1

Proposez un modèle RdP qui ne présente aucune situation de concurrence.

Question n°2

Combien de ressources sont utilisés ? Indiquer le rôle de chacune d'elles.

Question n°3

Le programme principal (voir l'annexe) commence par créer un processus fils et lui associe le traitement défini par la fonction `void coiffeur (void) ;`.

Le coiffeur gère la durée d'ouverture du salon par l'appel système `alarm()` et la durée de fermeture par l'appel système `sleep()`. Les durées sont prédéfinies par deux constantes (`DUREE_OUV` et `DUREE_FER`) exprimées en secondes

Le coiffeur n'est créé qu'au démarrage du programme. Par conséquent, le corps de la fonction sera constitué d'une boucle infinie.

A ce niveau on ne gère pas encore les clients, le coiffeur se contentera d'afficher son numéro de processus et le temps restant avant fermeture toutes les secondes.

Proposez une implémentation dans ce sens : création du coiffeur, mise en place de la fonction `coiffeur` avec gestion de l'alarme, de la durée de fermeture et de l'affichage. **Testez et validez votre solution** (Ne pas oublier de faire une capture d'écran pour le CR)**.**

Question n°4

L'objectif ici est de gérer la création des processus clients. Le programme principal met en place un mécanisme qui lui permet de créer un processus fils suite à la réception du signal `SIGUSR1`. Ainsi, on associera la fonction `void client (unsigned int numClient) ;` à chaque processus client créé.

A ce niveau on ne gère pas encore la communication entre le coiffeur et les clients. Le client se contentera donc d'afficher son numéro de client ainsi que le temps écoulé depuis son arrivée et ce toutes les 3 secondes.

Proposez une implémentation dans ce sens : mécanisme de création d'un client sur réception de `SIGUSR1`, mise en place de la fonction `client`.

Attention : le processus client doit bloquer le signal `SIGUSR1` dès sa création pour éviter qu'il se mette lui aussi à créer des clients sur réception de celui-ci. **Testez et validez votre solution** (Ne pas oublier de faire une capture d'écran pour le CR)**.**

Question n°5

Maintenant que vous disposez d'un processus `coiffeur` et de processus `clients`, vous allez mettre en place la communication entre le coiffeur et ces clients. Cette communication sera réalisée grâce aux **sémaphores nommés** (processus distincts) en conformité avec le modèle proposé à la question 1. **La création des sémaphores doit se faire avant la création d'aucun processus fils afin qu'il puissent les utiliser**.

Sachant que si toutes les chaises sont occupées, le client s'en va, combien de sémaphores doivent être créés.

Proposez une implémentation dans ce sens : création des sémaphores nommées. **Testez et validez votre solution** (Ne pas oublier de faire une capture d'écran pour le CR)**.**

Question n°6

Le processus coiffeur gère une variable locale indiquant le numéro du client traité et traite les clients dans l'ordre d'arrivée. Chaque client est coiffé en 5 secondes.

Adaptez le comportement du coiffeur pour la prise en compte des sémaphores et remplacez l'affichage par un affichage indiquant le numéro du client qui vient d'être coiffé.

Proposez une implémentation dans ce sens. **Testez et validez votre solution** (Ne pas oublier de faire une capture d'écran pour le CR)**.**

Question n°7

De même, Adaptez le comportement du client pour la prise en compte des sémaphores et remplacer l'affichage par un affichage indiquant le délai d'attente avant de se faire coiffer.

Proposez une implémentation dans ce sens. **Testez et validez votre solution** (Ne pas oublier de faire une capture d'écran pour le CR)**.**

Question n°8

Il s'agit maintenant de gérer l'acquittement des processus clients. En effet, quand un client a fini d'être coiffé, il se termine et le signale à son père (programme principal) qui doit se charger de sa terminaison.

Expliquez le mécanisme de terminaison d'un processus (signal, appel système).

Proposez une solution à base d'un thread auquel on associera la fonction `acquitterFils`. **Testez et validez votre solution** (Ne pas oublier de faire une capture d'écran pour le CR)**.**

Question n°9

Un CTRL-C, provoquera la terminaison de tous les processus en cours : coiffeur, clients en attente et le programme principal sans oublier de supprimer les sémaphores créés.

Proposez une solution permettant de terminer proprement ces processus.**Testez et validez votre solution** (Ne pas oublier de faire une capture d'écran pour le CR)**.**

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <signal.h>
#include <time.h>

#define CHECK(sts,msg) if ((sts) == -1) { perror(msg); exit(sts); }

#define N         3
#define DUREE_OUV 20
#define DUREE_FER (3 * DUREE_OUV)

unsigned int dureeOuv  = DUREE_OUV; // décompte durée d'ouverture
//unsigned int nbCL         = ???;  // nombre de chaises libres
unsigned int numClient = 0;         // numéro du client traité
//int pid [???];                    // tableau des clients actuels
void coiffeur (void) {
    // A compléter
}
void client (unsigned int numClient) {
    // A compléter
}

/*
void tuerLesFils(???)
{
    // Ce traitement est activé sur réception d'un CTRL C
    // Terminaison des sémaphores et de tous les processus
    // A compléter
}
*/

void * acquitterLesFils(void *arg)
{
    // Gérer la fin des processus fils à base de la signalisation fils/père
    // A compléter
    pthread_exit(0);
}

int main ()
{
    // Q5 : Mise en place des sémaphores
    //
    // Q3 : Mise en place du processus coiffeur avec la gestion de l'alarme
    //
    // Q4 : Installation du gestionnaire de SIGUSR1
    //
    // Q9 : Installation du gestionnaire du CTRL-C
    //
    // Q8 : Mise en place du thread d'acquittement
    while (1) {
        pause() ;
    }
    return EXIT_SUCCESS;
}
```