Coiffeur

Dans un salon de coiffure, il y a un coiffeur C, un fauteuil F dans lequel se met le client pour être coiffé et N sièges pour attendre.

Les événements possibles sont les suivants :

- S'il n'a pas de clients, le coiffeur C somnole dans le fauteuil F,
- Quand un client arrive et que le coiffeur C dort, le coiffeur se réveille et se lève. Le client s'assied alors dans le fauteuil F et se fait coiffer,
- Si un client arrive pendant que le coiffeur travaille :
    - si un des N sièges est libre, il s'assied et attend,
    - sinon il sort.
- Si un client quitte le salon et qu'un autre attend, celui qui attend prend le fauteuil F et se fait coiffer.

Fonctions pour le coiffeur : Coiffer ; (somnoler revient à mettre le processus en attente)
Fonctions pour les clients : RejoindreFauteuil ; Sortir ;

Clients et coiffeur sont représentés par des processus, le lancement d'un processus « client » correspond à l'arrivée d'un client dans le salon.

On supposera qu'un seul processus coiffeur peut-être lancé sur la machine.

Embarquement dans un autobus

La zone d'embarquement d'un arrêt de bus offre MAX places. Un usager qui arrive après le bus doit attendre le bus suivant.

Comment se fait l'embarquement :

- à l'arrivée du bus, les usagers déjà présents essaient de monter à bord. (Ceux qui se présentent après l'arrivée du bus doivent attendre le suivant).
- Lorsque tous les usagers qui attendaient sont montés à bord, le bus démarre,
- Si aucun candidat à l'embarquement n'est présent à l'arrivée du bus, celui-ci repart à vide.

Pour développer un simulateur de cette zone d'embarquement, on va développer le comportement des différents acteurs (passagers et bus) sous la forme de processus.

Nous supposons que deux fonctions sont à notre disposition pour développer ce simulateur :

- Embarquer, qui simule le fait qu'un passager monte dans le bus ;
- Partir, qui simule le départ du bus.

On suppose qu'un bus (respectivement un passager) arrive dans la zone d'embarquement dès que son processus commence à s'exécuter.

Première solution

Dans cette solution, chaque fois qu'un usager est monté à bord, il en fait monter un autre.

Deuxième solution

C'est le processus bus qui fait monter tous les passagers, l'un après l'autre, jusqu'au dernier présent à l'arrêt de bus.

Coiffeur

Initialisation des sémaphores et données partagées

Donnée partagée (sous forme d'un fichier partagé) :
Si la salle d'attente est pleine, on ne bloque pas les clients potentiels qui arrivent : ils sortent (fonction Sortir). On ne peut donc pas utiliser un sémaphore pour gérer le nombre de places dans la salle d'attente. On utilise donc une variable globale `NB_Attente` qui indique le nombre de clients dans la salle d'attente.
`NB_Attente = 0 ;`

Sémaphores :
`ClientPrésent` : bloque le processus coiffeur jusqu'à ce qu'un client entre dans le salon (le coiffeur somnole). `Init (ClientPrésent , 0).`
`ClientInstallé` : bloque le processus coiffeur jusqu'à ce qu'un client se soit installé sur le fauteuil. `Init (ClientInstallé , 0).`
`CoiffeurAFini` : bloque le processus client (celui en train de se faire coiffer) jusqu'à ce que le coiffeur ait fini son oeuvre. `Init(CoiffeurAFini, 0).`
`FauteuilCoiffeurLibre` : permet de s'assurer que :
(i) le client attend si le fauteuil n'est pas libre
(ii) le coiffeur est réveillé avant que le client ne rejoigne le fauteuil
`Init(FauteuilCoiffeurLibre , 0).`
`ExcMutAttend` permet d'assurer l'accès à la variable `NB_Attend` en exclusion mutuelle. `Init(FauteuilCoiffeurLibre , 1).`

Pseudo-code pour le programme COIFFEUR	Pseudo-code pour le programme CLIENT

`While(1){`
`  P(ClientPrésent) ;`
`  P(ExcMutAttend) ;`
`  V(FauteuilCoiffeurLibre) ;`
`  Nb_Attente-- ;`
`  V(ExcMutAttend) ;`
`  P(ClientInstallé) ;`
`  Coiffer() ;`
`  V(CoiffeurAFini) ;`
`}`
	`P(ExcMutAttend) ;`
	`if(Nb_Attente < N) {`
	`  Nb_Attente++ ;`
	`  V(ClientPrésent) ;`
	`  V(ExcMutAttend) ;`
	`  P(FauteuilCoiffeurLibre) ;`
	`  RejoindreFauteuil() ;`
	`  V(ClientInstallé) ;`
	`  P(CoiffeurAFini) ;`
	`  Sortir() ;`
	`} else {`
	`  V(ExcMutAttend) ;`
	`  Sortir() ;`
	`}`

Autobus : Première solution

Initialisation des sémaphores et données partagées:

Sémaphores :

`PlaceZoneEmbarquement` : assure qu'il n'y a pas plus de MAX usagers dans la zone d'embarquement. Ceux qui arrivent après attendent.
`Init(PlaceZoneEmbarquement, MAX) ;`
`AccesZoneEmbarquement` : l'accès à la zone d'embarquement est bloquée dès que le bus arrive. `Init(AccesZoneEmbarquement, 1) ;`
`AutorisationAMonter` : permet de bloquer un usager dans la zone d'embarquement jusqu'à ce qu'il puisse embarquer dans le bus.
`Init(AutorisationAMonter, 0) ;`
`AutorisationDémarrer` : permet de bloquer le bus jusqu'à ce que tous les usagers soient montés à bord. `Init(AutorisationDémarrer, 0) ;`

Donnée partagée :

`NB_Usagers` : indique le nombre d'usagers dans la zone d'embarquement.
`NB_Usagers = 0 ;`

Pseudo-code pour le programme BUS

```
P(AccesZoneEmbarquement) ;
if(NB_Usagers >0)
{
    V(AutorisationAMonter) ;
    P(AutorisationDémarrer) ;
}
Partir() ;
V(AccesZoneEmbarquement) ;
```

Pseudo-code pour le programme USAGER

```
P(PlaceZoneEmbarquement) ;
P(AccesZoneEmbarquement) ;
NB_Usagers++ ;
V(AccèsZoneEmbarquement) ;
P(AutorisationAMonter) ;
Embarquer() ;
V(PlaceZoneEmbarquement) ;
NB_Usagers-- ;
if(NB_Usagers==0)
    V(AutorisationDémarrer) ;
else
    V(AutorisationAMonter) ;
```

Autobus : Deuxième solution

Initialisation des sémaphores et données partagées:

Sémaphores :

`PlaceZoneEmbarquement` : assure qu'il n'y a pas plus de MAX usagers dans la zone d'embarquement. Ceux qui arrivent après attendent.
`Init(PlaceZoneEmbarquement, MAX) ;`
`AccesZoneEmbarquement` : l'accès à la zone d'embarquement est bloquée dès que le bus arrive. `Init(AccesZoneEmbarquement, 1) ;`
`AutorisationAMonter` : permet de bloquer un usager dans la zone d'embarquement jusqu'à ce qu'il puisse embarquer dans le bus.
`Init(AutorisationAMonter, 0) ;`
`ConfirmationMonté`: permet de bloquer le bus jusqu'à ce que tous les usagers soient montés à bord. `Init(ConfirmationMonté, 0) ;`

Donnée partagée :

`NB_Usagers` : indique le nombre d'usagers dans la zone d'embarquement.
`NB_Usagers = 0 ;`

Pseudo-code pour le programme BUS	Pseudo-code pour le programme USAGER

`P(AccesZoneEmbarqument) ;`
`int i ;`
`for(i=0 ; i<N ; i++) {`
`  V(AutorisationAMonter) ;`
`  P(ConfirmationMonté) ;`
`}`
`Nb_Usagers=0 ;`
`V(AccesZoneEmbarqument) ;`
	`P(PlaceZoneEmbarquement) ;`
	`P(AccesZoneEmbarquement) ;`
	`NB_Usagers++ ;`
	`V(AccèsZoneEmbarquement) ;`
	`P(AutorisationAMonter) ;`
	`Embarquer() ;`
	`V(PlaceZoneEmbarquement) ;`
	`V(ConfirmationMonté) ;`