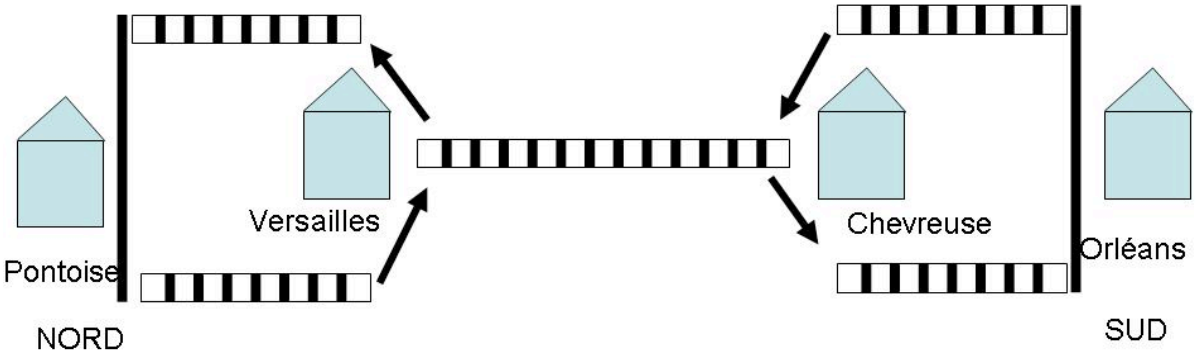
Exercice Moniteurs

Compétition d'accès pour deux classes de processus



Les règles de circulation sur la voie unique sont les suivantes :

1. Le tronçon ne doit jamais être emprunté simultanément par deux trains allant en sens inverse ;
2. le tronçon peut être emprunté par un ou plusieurs trains allant tous dans le même sens ;
3. il n'y a pas de sens de parcours prioritaire.

Pour étudier la rentabilité de cette ligne, on désire effectuer une simulation du trafic, pour cela, on introduit deux classes de processus : les trains PONTOISE-ORLEANS et les trains ORLEANS-PONTOISE, qui se décrivent comme suit :

Tâche PONTOISE-ORLEANS :

Début

Parcours (PONTOISE, VERSAILLES) ;
Entrée_nord ;
Parcours (VERSAILLES,CHEVREUSE) ;
Sortie_sud ;
Parcours (CHEVREUSE, ORLEANS) ;

Fin

Tâche ORLEANS-PONTOISE:

Début

Parcours (ORLEANS, CHEVREUSE) ;
Entrée_sud ;
Parcours (CHEVREUSE, VERSAILLES) ;
Sortie_nord;
Parcours (VERSAILLES, PONTOISE) ;

Fin

Question :

On considère le cas où toute coalition est autorisée.

Écrire un moniteur contenant les 4 points d'entrée **Entrée_nord , Sortie_sud, Entrée_sud , Sortie_nord**

On pourra utiliser les conditions Accès_Nord, Accès_Sud et des variables d'état NS, SN nombres de trains sur le tronçon dans chaque sens, AttNS, AttSN nombre de trains en attente dans chaque sens.

D'autres solutions sont possibles !

Type Troncon = **moniteur**
{Variables locales}

Var NS,SN, AttNS, AttSN : **entier** ;

Accès_Nord ; Accès_Sud : **condition** ;

{points d'entrée}

Procédure Entry Entrée_Nord ;

Début

Si SN>0 alors AttNS=AttNS+1 ; ***Accès_Nord.*Wait** ; finsi ;

{blocage si trains dans l'autre sens}

NS=NS+1 ;

Si AttNS>0 alors AttNS=AttNS-1 ; ***Accès_Nord.*Signal** ; finsi ;

{si des trains sont en attente, alors réveil du prochain train en attente ; le premier train en attente est réveillé par le dernier train SN}

Fin

Procédure Entry Sortie_Sud ;

Début

NS=NS-1 ;

Si NS==0 alors {dernier train NS}

Si AttSN>0 alors AttSN = AttSN -1 ; ***Accès_Sud.*Signal** ;

{si des trains SN sont en attente, alors réveil du premier train }

finsi ;

finsi ;

Fin

Procédure Entry Entrée_Sud ;

Début

Si NS>0 alors AttSN=AttSN+1 ; ***Accès_Sud.*****Wait** ; finsi ;

{blocage si trains dans l'autre sens}

SN=SN+1 ;

Si AttSN>0 alors AttSN=AttSN-1 ; ***Accès_Sud.*****Signal** ; finsi ;

{si des trains sont en attente, alors réveil du prochain train en attente ; le premier train en attente est réveillé par le dernier train NS}

Fin

Procédure Entry Sortie_Nord ;

Début

SN=SN-1 ;

Si SN==0 alors {dernier train SN}

Si AttNS>0 alors AttNS = AttNS -1 ; ***Accès_Nord.*****Signal** ;

{si des trains NS sont en attente, alors réveil du premier train }

finsi ; finsi ;

Fin

Début

{initialisations des variables locales}

NS=0; SN=0 ; AttNS=0 ; AttSN=0 ;

Fin