```
/*
 * Classe spécifique à la gestion globale de la barrière
 * Possibilité de contrôler la situation du véhicule
 * par rapport à la borne d'accès, ainsi que la position de la barrière
 */

#ifndef _BARRIERE_H
#define _BARRIERE_H

#include <Socket.h>
#include <string.h>

//-----------------------------------------------------------------
enum Position {enHaut, auMilieu, enBas};
enum Situation {enEntree, enDessous, enSortie, aucune};
//-----------------------------------------------------------------
class Barriere
{
  Socket &barriere;
public:
  Barriere(Socket &service) : barriere(service) {}
  void setCommandeSensMontee();
  void setCommandeSensDescente();
  bool isInFDCH();
  bool isInFDCL();
  bool isBoucleAmont();
  bool isBoucleAval();
  Position getPosition();
  Situation getSituation();
};
//-----------------------------------------------------------------


// Cette méthode permet d'activer la montée de la barrière.
// Si la barrière est déjà en haut, aucune action n'est réalisée.
inline
void Barriere::setCommandeSensMontee()
{
  barriere.envoyer("Barriere:setCommandeSensMontee");
  barriere.recevoir();
}

// Cette méthode permet d'activer la descente de la barrière.
// Si la barrière est déjà en bas, aucune action n'est réalisée.
inline
void Barriere::setCommandeSensDescente()
{
  barriere.envoyer("Barriere:setCommandeSensDescente");
  barriere.recevoir();
}

// Indique l'état du capteur "Fin De Course Haut"
// permettant de savoir si la barrière est en position haute.
inline
bool Barriere::isInFDCH()
{
  barriere.envoyer("Barriere:isInFDCH");
  return strcmp(barriere.recevoir(), "haut") == 0;
}
```

```
// Indique l'état du capteur "Fin De Course Bas (Low)"
// permettant de savoir si la barrière est en position basse.
inline
bool Barriere::isInFDCL()
{
  barriere.envoyer("Barriere:isInFDCL");
  return strcmp(barriere.recevoir(), "bas") == 0;
}

// Indique l'état du capteur de présence de la voiture en entrée.
inline
bool Barriere::isBoucleAmont()
{
  barriere.envoyer("Barriere:isBoucleAmont");
  return strcmp(barriere.recevoir(), "entree") == 0;
}

// Indique l'état du capteur de présence de la voiture en sortie.
inline
bool Barriere::isBoucleAval()
{
  barriere.envoyer("Barriere:isBoucleAval");
  return strcmp(barriere.recevoir(), "sortie") == 0;
}

// Précise la position de la barrière.
inline
Position Barriere::getPosition()
{
  if (isInFDCH()) return enHaut;
  if (isInFDCL()) return enBas;
  return auMilieu;
}

// Précise la situation du véhicule pa rapport à la borne d'accès.
inline
Situation Barriere::getSituation()
{
  bool entree = isBoucleAmont();
  bool sortie = isBoucleAval();
  if (entree && sortie) return enDessous;
  else if (entree) return enEntree;
  else if (sortie) return enSortie;
  return aucune;
}

#endif
```