Nerea Lefa Práctica 2 PRPA.

monitor C){

nc1 int = 0 number of cars 1
nc2 int = 0 -"- cars 2
nh int = 0 -"- humans
ncw1 int = 0 number of cars 1 waiting
ncw2 int = 0 -"- cars 2 "
nhw int = 0 -"- humans "
turn int = 0 v 1 v 2
condition nocars, nobody1, nobody2

↓ cars 2

turns
0. humans
1 cars 1
2. cars 2

↑ cars 1

Invariante del monitor

*cambiar los delays

nh ≥ 0 nhw ≥ 0
nc1 ≥ 0 nc1w ≥ 0
nc2 ≥ 0 nc2w ≥ 0

nh > 0 → nc1 == 0 ∧ nc2 == 0 ∧ turn = 0

nh = 0 → nc1 > 0 ∨ nc2 > 0

nocars → nh > 0 ∧ (¬nobody1 ∧ ¬nobody2)

¬nocars → nh = 0 ∧ (nobody1 ∨ nobody2)

nc1 > 0 → nc2 == 0 ∧ nh == 0 ∧ turn = 1

nc2 > 0 → nc1 == 0 ∧ nh == 0 ∧ turn = 2

Operations

```
want_walk()
    nhw = nhw + 1
    nocars.wait( nc1==0 ∧ nc2==0 )
    nhw = nhw - 1
    nh = nh + 1
```

∧(turn==0 ∨ ncw1==0 ∨ ncw2==0)

```
stop_walking()
    nh = nh - 1
    turn = 1 ∨ 2
    if nh == 0 :
        nobody1.notify()
        nobody2.notify()
```

```
want_drive1()
    ncw1 = ncw1 + 1
    nobody1.wait( nh==0 ∧ nc2==0 )
    ncw1 = ncw1 - 1
    nc1 = nc1 + 1
```

∧(turn==1 ∨ nhw==0 ∨ ncw2==0)

```
stop_driving()
    nc1 = nc1 - 1
    if nc1 == 0 :
        nocars.notify()
        nobody2.notify()
```

turn = 0

```
want_drive2()
    ncw2 = ncw2 + 1
    nobody2.wait(nh==0 ∧ nc1==0)
    ncw2 = ncw2 - 1
    nc2 = nc2 + 1
```

∧ (turn==2 ∨ nhw==0 ∨ ncw1==0)

```
stop_driving2()
    nc2 = nc2 - 1
    if nc2 == 0 :
        nocars.notify()
        nobody1.notify()
```

turn = 0

Processes

human	car 1
CH.want_walk()	CH.want_drive1()
passing	passing
CH.stop_walking()	CH.stop_driving1()

car 2

CH.want_drive2()

passing

CH.stop_driving2()

• El puente es seguro

(no hay coches y peatones a la vez en el puente
no hay coches en sentidos opuestos)

Si hay peatones, es decir, $nh > 0$ quiero demostrar que $nc1 == 0 \wedge nc2 == 0 \wedge turn = 0$

En principio esto es cierto por la definición del monitor.
Proceso: human
Al ejecutar want_walk(), el noCars.wait garantiza que nh no se va a aumentar si la condición en el paréntesis es falsa.
Después, cuando stop-walking se ejecuta, el if-stateuente garantiza que los procesos car1 y car2 se notifican solamente si $nh == 0$, es decir si no hay peatones en el puente.

Cuando hay coches en el sentido 1, por ejemplo, $nc1 > 0$ quiero demostrar que $nc2 == 0 \wedge nh == 0$ es decir no hay coches en el sentido contrario ni peatones.
Al ejecutar want-drive1, la condición del "wait" es $nh == 0 \wedge nc2 == 0$. Si esto es el caso, nc1 se aumenta. En el "stop-driving" los notificadores se ejecutan

si nc1==0, o sea, si no hay coches en el sentido 1 en el puente.
Analogamente para los coches del sentido 2

- Ausencia de deadlocks. todos están bloqueados ningún entra

Si todos están bloqueados, esto significa que
nh=0, nc1=0, nc2=0 y nhw>0, nc1w>0, nc2w>0
y todos están bloqueados en el wait
Eso no puede occurrir por la variable turn
que siempre tiene un valor (0 v 1 v 2).
Inicialmente, arbitrariamente se asigna un valor.
Si, por ejemplo, turn = 1, los coches en el sentido 1 pueden pasar el puente.
Despues de pasarlo, turn = 0, para que lo pasen los peatones.
Si no hay peatones, la condición del wait es cierta por "nhw = 0".

• Ausencia de inanición

~~Un proceso~~
~~Si un proceso human está bloqueado indefinidamente,~~
~~entonces~~

Si un proceso human está en inanición, entonces está bloqueado indefinidamente en nocars.wait, es decir nocars.notify() no se ejecuta o la condición en el "wait" nunca es cierta.

Si nc1 > 0, entonces want_drive1 están ejecutando y stop_driving1

Por la desminución del nc1, del ncw1 y por la asignación turn=0, en algun momento se ejecutará nocars.notify() y la condición del wait será cierta.

Analogamente para los coches