```python
"""
Solution to the one-way tunnel
"""
import time
import random
from multiprocessing import Lock, Condition, Process, Manager

SOUTH = 0
NORTH = 1

NCARS = 5

class Monitor():
    def __init__(self):
        self.manager = Manager()
        self.mutex = Lock()
        self.inTunnel =  self.manager.list([0,0])
        self.semaphore= Condition(self.mutex)
        self.carDir= None

    def car_dir(self, car_direction):
        self.carDir = car_direction

    def validTunnel(self):
        return  (self.inTunnel[(self.carDir +1)%2] == 0)

    def wants_enter(self, car_direction):
        self.mutex.acquire()
        self.car_dir(car_direction)
        self.semaphore.wait_for(self.validTunnel)
        self.inTunnel[car_direction] += 1
        self.mutex.release()

    def leaves_tunnel(self, car_direction):
        self.mutex.acquire()
        self.inTunnel[car_direction] -= 1
        self.semaphore.notify()
        self.mutex.release()

def delay(n=3):
    time.sleep(random.random()*n)

def car(cid, direction, monitor):
    print(f"car {cid} direction {direction} created")
    delay(6)
    print(f"car {cid} heading {direction} wants to enter")
    monitor.wants_enter(direction)
    print(f"car {cid} heading {direction} enters the tunnel")
    delay(3)
    print(f"car {cid} heading {direction} leaving the tunnel")
    monitor.leaves_tunnel(direction)
    print(f"car {cid} heading {direction} out of the tunnel")

def main():
    monitor = Monitor()
    cid = 0
    cars=[]
    for _ in range(NCARS):
        direction = NORTH if random.randint(0,1)==1  else SOUTH
        cid += 1
        p = Process(target=car, args=(cid, direction, monitor))
        p.start()
        cars.append(p)
        time.sleep(random.expovariate(1/0.5)) # a new car enters each 0.5s

    for c in cars:
        c.join()

if __name__ == "__main__":
 main()
```

INVARIANTE

Self. inTunnel = [a,b] ; a,b>=0 and (Si a>0 -> b=0) and (si b>0 -> a=0)

La inicialización se cumple self.inTunnel =[0,0]

Wants enter:

Supongamos que se cumple el invariante y tomamos car_direction =1 (análogo para car_direction = 0).

Si cumple validTunnel -> inTunnel = [0,b] con b>=0

Entonces inTunnel = [0,b+1]

Y se sigue cumpliendo el invariante

Leaves tunnel:

Renombramos b' = b+1

Supongamos b' >0

Cuando sale el coche ¿b' -1 >0? Sí porque b'-1 = b+1-1= b>0 y esto es cierto por el invariante (b>=0)

Entre wants enter y leaves tunnel se pued haber actualizado la b' pero siempre cumpliendo el invariante (b>=0)