Operating Systems 2010-2011 – CONCURRENCY

Write two implementations in C language, one that uses the semaphores and one that uses monitors, representing a concurrency solution to this problem:

Parachute jumps

The Cartoonia airport organizes parachute jumps.

- The airport provides a shuttle to accompany the would-be parachutists from the boarding gate to the track where the plane "FANDANGO" (FG) lands and take off.
- The shuttle can accomodate a maximum of M passengers.
- The N potential parachutists (N>M) arriving at the gate for boarding, try to get on the shuttle to reach the plane.
- In the case there are no passengers the shuttle is suspendend, waiting a passengers who wants to reach FG. In the case in which all the seats of the shuttle are filled, would-be parachutists are suspended, waiting free seats on the shuttle and walking in the hall in front of the gate.
- The shuttle leaves when at least one passenger is on board, reach the track, drop the passenger/s and return back to the boarding gate where it will wait at least a passenger.
- FG has a P maximum passengers (N>P).
- If FG is full, visitors are suspended on the track, waiting for GH returns to pick up new passengers.
- FG leave only when one of the following condition is true:
  - At least one passenger is on board and there are no other passengers waiting on track
  - when has embarked P passengers
- If there are free seats on FG, then would-be parachutists line up to withdraw a parachute from the attendant who is on track, otherwise they wait looking at the sky above the airport.
- If there are vacancies and if they took a parachute, would-be parachutists board FG for the launch.
- Initially there are P parachutes.
- At the end of each flight of FG, a janitor collects the parachutes from passengers were thrown, folds and return them to the attendant on the track, so he can give them to the new would-be parachutists.
- Initially, the shuttle and FG are free, all parachutes are avaiable at the attendant on the track and there are not waiting passengers.

The proposed solution must ensure the maximum degree of parallelism, must not use busy waiting, and must not generate deadlock or starvation. Comment clearly the code.