ARP 2021 - THIRD ASSIGNMENT

The third assignment is cooperative. It is solved by super-groups of about 10-12 students, which are unions of 5- 6 already active groups. However, these numbers are not strictly compulsory, and may vary. The assignment is thought to use already developed codes and skills. Differently from assignments 1 and 2 *all* students will upload their codes. In the following we will consider the case of a super-group composed by 6 groups of two students.

Aim

To develop **a concurrent program composed by 6 processes**, communicating through sockets. Each student must have a running copy of the program; each student has a source of her/his code (developed in her/his group), and *only* the executables given by the other groups involved in the super-group. Hence, every student runs all 6 processes, of which 5 are directly connected through sockets at run level (no compilation nor link).
The aim is to learn how several professionals develop parts of a whole bigger project which run together, connecting in *plug-and-play* way. The different groups should not be aware of how the party groups implemented their codes; all must agree only the *connection protocols* of the processes.

We strongly push you to act as described above. We cannot exclude that one group copies the code of some other group (as we will see, 5 processes over 6 are similar). In this case you will have less than what the university tuition fee could give you back... so, play seriously this game, and *exchange only executables among groups*. This will spre time and make much simpler the individual work at group leve,. However, we will read your codes.

The program

The program developed by a super-group is composed by 6 processes:

(a) 5 drones exploring autonomously a rectangular area
(b) 1 masterstation showing the drones positions and preventing their collision

drones

Each drone starts from a point in the working area, travels randomly and consume power. Initial positions may be fixed. After total power consumption they stay still ("landing"). You should inspire yourself / re-use the code of one motor of the assignment 2, acting in two coordinates instead of one at any step. The only difference is to vary the direction after n elementary steps (you decide n) to generate reasonably long trajectories and avoid "vibrating" around the running point.
Before moving, the drone asks the master for permission. If the permission is denied, the drones refrain from moving.

master

The master receives all requests by drones to move, checks if a movement produces a collision or exit from the work area. In that cases it denies the motion.
The master prints the positions of drones in a normal *ASCII* shell, in a rectangle of 40 lines by 80 characters. It simply refreshes the whole printout at maximum rate which reasonably keeps a still image of the whole working area.
Still and operating drones must be distinct in a simple way in the printout.

improvements

Many voluntary improvements may be done, basing on your skill, your time, your satisfaction. For example:

i. refueling after landing
ii. smart drones (power, speed, exploration strategy..)
iii. visualization of the already explored area
iv. coverage algorithm
v. interactive commands by the master
vi. vertical motion
vii. .....