UNIVERSIDAD DE MÁLAGA
Dpto. Lenguajes y Ciencias de la Computación

Systems Programming and Concurrency

ANDALUCÍA TECH
Campus de Excelencia Internacional
Campus of International Excellence

Exam of Java, June 2022

Description of the system

A residential building has **N** lifts available to the public, numbered from 0 to N-1. After a party in the city centre, **V** persons arrive to the building to go their homes. The lifts are very small and each of them has a capacity for only one person. We want to model the process in which each **Person** waits for an available lift in the **Building**, takes it, uses it and releases it making it available again. Let's model the system with the next classes:

- **Person**. A class that inherits from Thread and models a person that wants to take a lift to go home; once inside the lift, a time elapses until the lift reaches his/her floor; then s/he releases the lift that, hence, is available again to be used by another person.
- **Building**. This class models the shared resource. It holds a number that represents the number of lifts and has the next methods that are used by the Persons:
  - `int boardOnLift(int id)`: a person identified with **id** uses this method to come into any lift. He waits into this method until any lift is available and, then, returns the **id of the lift** assigned to him/her.
  - `void boardOffLift(int id, int liftId)`: a person identified with **id** uses this method to express that s/he has arrived to his/her floor and, hence, s/he releases the lift making it immediately available to any other person.
  - `void showUsage()`: this function displays how many times has been used each lift. At the beginning, each lift has been used 0 times and, each time a new person uses a lift (let's say lift **id**), we increment the usage of the corresponding lift.
- In addition, there is class **Driver** that creates a Building instance (with **N** lifts) and **V** Person instances. This class has to call to `showUsage()` when every person is at his/her home, i.e., all the threads have finished their execution.

You have to develop two implementations of the class Building:

1. Binary semaphores (4 points).
2. Monitors or Locks (4 points).

In addition, you have to complete the function **main** in order it to call to `showUsage()` when all the threads are finished (2 points).

The next is a trace of the output of the program:

```
Person 0 takes the lift 0
Person 5 takes the lift 1
Person 4 takes the lift 2
Person 5 releases the lift 1
Person 6 takes the lift 1
Person 0 releases the lift 0
Person 2 takes the lift 0
Person 2 releases the lift 0
Person 3 takes the lift 0
Person 4 releases the lift 2
```

```
Person 7 takes the lift 2
Person 7 releases the lift 2
Person 1 takes the lift 2
Person 3 releases the lift 0
Person 8 takes the lift 0
Person 6 releases the lift 1
Person 9 takes the lift 1
Person 9 releases the lift 1
Person 10 takes the lift 1
Person 1 releases the lift 2
Person 11 takes the lift 2
Person 8 releases the lift 0
Person 12 takes the lift 0
Person 12 releases the lift 0
Person 13 takes the lift 0
Person 13 releases the lift 0
Person 14 takes the lift 0
Person 14 releases the lift 0
Person 15 takes the lift 0
Person 10 releases the lift 1
Person 16 takes the lift 1
Person 11 releases the lift 2
Person 17 takes the lift 2
Person 15 releases the lift 0
Person 18 takes the lift 0
Person 16 releases the lift 1
Person 19 takes the lift 1
Person 17 releases the lift 2
Person 18 releases the lift 0
Person 19 releases the lift 1
Lift 0 used 9
Lift 1 used 6
Lift 2 used 5
```