Fanti Andrea 235808

Lab 11 exercise 02

This exercise is very similar to number 1 but there is a cycle: after D terminates its execution A has to execute again its cycle.

To implement this feature a semaphore is used: this semaphore (`sems[0]`) is locked by thread A and unlocked by thread D.

Each thread contains a `for` loop to execute its core 10 times.

Notice that here we need to use 4 semaphores in total (**2** more than exercise 1) because thread B and C must be unlocked by 2 different semaphores, otherwise it is possible that a thread finishes its cycle and start another one right away (e.g an execution order of A B B D or A C C D is possible).

It is worth noting that once thread D finishes its execution it is possible that there are still some thread executing, if the scheduler removed the control from (for instance) C after it unlocked `sems[3]` for the last time but before its `pthread_exit` is executed.