SDP 2016/17 - Lab 1 - Davide Gallitelli S241521

E02 - Synchronization with semaphores

The second part of the first lab aims at implementing a C server/client application by means of the *pthread* library. Its purpose is to read the two binary files previously generated, fv1.b and fv2.b. Each time a value is read from a file by a thread and stored in the global variable *g*, the server multiplies that value by 3 and notifies the thread which wrote *g* to print on the console.

In order to do so, three sempahores are to be used. The first one, sempahore s1, is the one that allows client threads to run. As only one thread is supposed to be running at a time, both threads wait on the same semaphore, which is initialized to 1 - only one can enter the critical section. The first CS generates the value for *g* by reading the next int from the file descriptor specified as parameter of the thread until EOF is reached (*read()* returns false, condition of the *while* is not met). Once the value is read, the semaphore s2 signals the server to do its computation - multiplication by 3 - and increase its counter of requests served. The semaphore s3 allows the thread which just generated the value to print the new computation, and then unlock any other thread waiting for s1 semaphore.

The value *g = -1* is used as reference value for the server to understand when the thread is done reading: the computation section of the server keeps running until all clients have processed their files - represented in the code by the counter *myChildCount* being lower than the constant *CHILDREN_NO*.

By using the *CHILDREN_NO* constant, this program is extremely versatile and allows for any number of clients to be launched, provided enough files (CHILDREN_NO files, each named *fvX.b*, with X from 1 to *CHILDREN_NO + 1*).