LAB – C2

SISTEMES OPERATIUS

Bru Gladiador

Rubén Gómez

ÍNDEX

Tabla de contenido

[Activity 1: Processes, shared memory and semaphores 2](#_Toc129346694)

[Compilar i Executar 2](#_Toc129346695)

[Activity 2: Processes and Named pipes 4](#_Toc129346696)

[Activity 3: Threads and mutexes 5](#_Toc129346697)

# Activity 1: Processes, shared memory and semaphores

To make this activity we need to create two processes inside a program named activity2.c, one shared memory space, and two named semaphores.

* To create the shared memory space, you can use the **shm\_open** function call.
* To limit the space to 4 bytes you can use the **ftruncate** function call.
* To map the memory space in the process, you can use the **mmap** function.

The semaphores will be named "/activity2\_sem1" and "/activity2\_sem2” and will be used to indicate each process’s turn to write to shared memory.

* To create and initialize semaphores, you can use the **sem\_open** and **sem\_init** calls.
* To get and release the semaphores, you can use the **sem\_wait** and **sem\_post** calls.
* Finally, remember to release the semaphores with **sem\_close**.

## Compilar i Executar

First of all, we need to open a terminal in the directory where we have the "activity2.c" file.

Next we are going to enter the following command to compile the source code:

gcc -o activity2 activity2.c -lrt -pthread

This command compiles the "activity2.c" file and creates the "activity2" executable file. The "-lrt" option is necessary to use the functions related to shared memory, and the "-pthread" option is necessary to use the functions related to semaphores.

Once the program has been compiled successfully, we are going to execute with the following command:

./activity2

This command runs the "activity2" file and starts the simulation of the processes that communicate through shared memory and semaphores.

Additionally, after each execution, it is important to free the resources used by the program through the cleanup commands that are at the end of the code.

# Activity 2: Processes and Named pipes

# Activity 3: Threads and mutexes