Operating Systems

Lab 11 Exercise – Shell, Pthread semaphores and mutexes

Learning goals: this laboratory activity is devoted to the use of shell and synchronization system calls.

Exercise 1

Implements a concurrent C program, **using semaphores** (not a pipe), that generates a producer and two consumer threads, and waits for their termination. The producer threads produces, at random intervals of 200 milliseconds, **`10000`** integer numbers (from **`0`** to **`9999`**), and puts each of them in a shared circular buffer of dimension **`BUFFER_SIZE`** (e.g., equal to **`16`**), and finally puts **`-1`**. Then it exits.

Each consumer thread gets a number at a time from the shared buffer, and writes it on a file **`out_TID.txt,`** where **`TID`** is the thread number.

If the read number is **`-1`**, this number is not written on the file, instead the thread puts in the shared buffer another **`-1`**, to allow the second thread to read it, and exits.

Test, with a **`bash`** script, that each file **`out_TID.txt`** contains numbers that have been received in the correct sequence, i.e., in ascending order, and that the two files contain all the number between **`0`** to **`9999`** .

Exercise 2

Recalling the definition of a general semaphore, write the C functions **`s_init`**, **`s_wait`**, and **`s_post`** that implement a general semaphore by means of **`mutexes`** using the appropriate data structure

Test your functions in your Producer & Consumer solution of the previous exercise replacing the Pthread **`sem_init`**, **`sem_wait`**, and **`sem_post`** calls with **`s_init`**, **`s_wait`**, and **`s_post`**.

Exercise 3

Write a Bash script that compares the contents of two directories, including both files and sub-directories. The names of the two directories are passed as arguments in the command line.

The script must, check that the arguments are directories, and it must write to **`stdout`** the list of files and sub-directories that don't appear in both directories.

Summary

At the end of this laboratory activity, you should became more familiar with **`bash`** and with semaphores and mutexes.