Fundamentos de Sistemas de Operação

MIEI 2018/2019

Homework Assignment 3

Deadline and Delivery

This assignment is to be performed by groups of **2 students** – any detected frauds will cause failing the discipline. The code must be submitted for evaluation via the Mooshak system (`http://mooshak.di.fct.unl.pt/~mooshak/`) using group's account -- the deadline is ***19h00, December 16th, 2018 (Sunday)***.

Description

Remember the program from laboratory session 10 (and 9). Your program counts the number of times a specified number appears in an array (vector) of integers using processes created by *fork( )* that return partial results through pipes. In this assignment you should change the code produced in lab session 10 in two aspects:

1. The main process (the one that launches all the workers) should get the results from the child processes in a different way. If *N* workers are launched, workers may be numbered form *0* to *N-1*. In lab10 the main process performed the loop

   ```
   for ( int p=0; p<nprocs; p++ ) {
           int c;
   // TODO: read results from all workers
           count += c;
   }
   ```

   where it waits and receives first the reply of worker 0, then the reply of worker 1, and so on.

   You should change the code to guarantee that the main process receives the replies as soon as they are available. This means that it will receive first the reply of the fastest process, then the response of the 2nd fastest and so on. This is possible if the main process uses the system call *select( )* to wait for any result to be available. For details of the use of this system call please see the slides of the 7th of December class, and chapter 33 of the book *Operating Systems: Three Easy Pieces* available from *http://pages.cs.wisc.edu/~remzi/OSTEP/threads-events.pdf*

2. The initial process should write to *stdout* the response received from each worker with the worker's number (between *0* a *N-1*). This number corresponds to its launch order and respective index in the pipes array (see code). So, the first printed line should have the identification of the worker that replied first, the second line the worker that replied second, and so on. Take into account that a worker may abort (eg. if there is not enough memory), closing its pipe, without sending any data.

All your implementation should be carried in the file `process.c` and this is the only file to submit to the mooshak system. Use the `printf` provided in the source code to notify of each received data or of any prematurely closed pipe.

Testing

To test your solution, you may use file `countNprocMain.c` that launches a set of processes to perform the counting. To guarantee that each worker process takes a distinct time to reply, for testing proposes each worker sleeps for a random number of milliseconds before replying.