Project tasks' description
Part 3^{nd}
Processes in Linux

Mateusz Cholewiński

28.03.2022

1 Document goal

Goal of this document is to define tasks to be done during the Real-Time Operating System project classes.

2 Scope of document

Scope of this document is limited only to tasks to be realized in Linux system in Virtual Machine or normal PC.

3 Realization, documentation and rules

Task should be done in such way to meet all following requirements:

1. During implementation put some meaningful comments.
2. During implementation use meaningful names for structures, variables etc.
3. Software development should be done based on Makefile file.
4. Solutions regarding Linux application shall be based on POSIX.
5. Prepare task solution documentation in the form of **report**, which will consist of crucial source code parts, examples, description of implementation, algorithms and assumptions. Such documentation should be free from grammar and language errors.

4 Tasks

1. (20 pts) Create two applications, connected with a pipe. Data written by user, via console, should be sent to other application and should be printed on other console.

2. (40 pts) Propose a solution for Readers and Writers problem (described in previous list of tasks) based on shared memory concept. Some general informations are described below:

   - use one reader and one writer,
   - reader is waiting for sync signal triggered when shared memory has been updated,
   - writer is reading data from text file (e.g. `https://ocw.mit.edu/ans7870/6/6.006/s08/lecturenotes/files/t8.shakespeare.txt`),
   - writer is writing some big number of data (e.g. 1000 chars) to shared memory in one run,
   - reader is reading data and writes it to file, other than reader uses,
   - both files shall look the same after run,
   - name of files might be hard coded or read as an execution argument.

3. (40 pts) Estimate a value of π using a Monte Carlo Method `https://academo.org/demos/estimating-pi-monte-carlo/`. Use a posix queues as a method of communication.

   - there should be one server process, which will create, using a fork() computing processes,
   - server process should also gather results from child processes and compute a mean value and communicate it to user,
   - server process should take a child process number and number of points from user and share with child processes,
   - child processes should compute values and after the computations are

     finished, send it to server.

5 Deadline

According to Teacher's decision.