CITS2002 Systems Programming 2020, 1st Project Marking Sheet

Submission of **CLEARY, THOMAS (21704985)**

Marks awarded from manual code inspection:

• Clear, descriptive, and sufficient comments	2 / 2
• Use of #defines for constants, rather than embedded numbers	2 / 2
• Consistent layout and regular indentation of code	2 / 2
• Choice of meaningful identifier names	2 / 2
• Design of data-structures to store process, and pipe information	4 / 4
• Initialising data-structures while reading the eventfile (just once)	2 / 2
• Data-structures and functions to manage Ready queue, states, events	2 / 2
• Control-flow within, possibly new functions supporting, run_simulation()	3 / 4

Fantastic submission (hopefully it passes all of the automated tests). A couple of small pieces of feedback:

+ Unsure why a pointer to the current runnning process is needed (could you not just use an index or a struct variable instead? Did not deduct a mark for this as it is a small issue but does impact on readability a bit)

+ It is inefficient to always be iterating over MAX_PROCESSES (and likely MAX_SYSCALLS_PER_PROCESS). For example, instead of checking isProcessActive() every iteration of your pipSim() outer while loop (for this I also don't think that you should have PIDs that are not in your eventfile listed as NEW) you could have a variable that keeps track of how many processes are active in your system at any one time (increment this on fork, decrement on exit) OR you could keep track of how many processes were found in your eventfile (and decrement this on exit). Checking this variable would be more efficient than continually running isProcessActive().

Any other comments:

Please review the sample solution -
https://teaching.csse.uwa.edu.au/units/CITS2002/projects/project1.php

---

TOTAL MARKS FROM MANUAL CODE INSPECTION: **19 / 20**

TURN PAGE FOR MARKS AWARDED FOR TESTING PROJECT

CITS2002 Systems Programming 2020, 1st Project Marking Sheet

Submission of **CLEARY, THOMAS (21704985)**

Marks awarded from compiling submission and running a series of tests:

• Program compiled without warnings or errors	2 / 2
• Successful - A single process exits immediately	2 / 2
• Successful - A single process computes (for less than the timequantum), then exits	2 / 2
• Successful - A single process computes (for multiple timequanta)	2 / 2
• Successful - Parent forks new child, parent and child both execute	2 / 2
• Successful - Parent forks two child processes, all execute	2 / 2
• Successful - Parent forks child process, child executes, parent waits for child	2 / 2
• Successful - Initial process sleeps, computes, sleeps, computes, and then exits	2 / 2
• Successful - Process creates pipe, forks child, parent writes to pipe, child reads from pipe	2 / 2
• Successful - Process creates pipe, forks child, parent writes to pipe, child reads 4x from pipe	2 / 2

TOTAL MARKS FROM AUTOMATED TESTING: **20 / 20**

PROJECT TOTAL: 39 / 40