CSCI 3232 Systems Software Assignment 8

Upload all your files to the correct dropbox in Folio before the deadline --- **11:30PM Jul 18, Sunday, 2021.**

**Note: Do not start to attempt this homework until you have practiced and understood all relevant sample codes in slides and Folio’s example codes. If Folio complains, you can add .txt extension to the submitted files. For example, makefile.txt.**

1. (18 pts) Write a C program A8p1.c that accepts one command line argument that is assumed to be a string of lower case English characters. Call the ***fork*** function to produce two processes. In the child it should print out the original version of the string. In the parent it should print out both the original and the complementary version of the string on two separate lines, i.e. ‘a’→’z’, ‘b’→’y’, ‘c’→’x’,…, ‘z’→’a’ (e.g. child: “abc”; parent: “abc” and “zyx”). Specify in the output whether the parent or child process is printing. Submit source file, not screen shots.
2. (40 pts) Follow the example programs unix\_pipe.c, named\_pipe.c and shm-posix-combined.c to write **three** versions (two pipe versions and one shared memory version) of an interprocess communication program (A8p2\_unixpipe.c, A8p2\_namedpipe.c and A8p2\_shm.c) in C. Each version should create two processes using ***fork***. The child should send or share twenty random integers *a1,…,a20* in the range from -5 to 19 inclusive to the parent. The child should print out the values of these integers (ten integers per line). The parent should decide and print out whether the two vectors (a1,…,a10) and (a11,…,a20) are orthogonal. For the shared memory version, do not call the *shmget* function. Sample runs of the programs in 1 & 2 can look like the following. You do NOT need to submit screen shots. Instead submit source code files.

[kwang@computer][~/temp]$ ./A8p1 abc

I am the child. Printing the original version:

abc

I am the parent. Printing the original and complementary version:

abc

zyx

[kwang@computer][~/temp]$ ./A8p2\_shm

child wrote into the shared memory:

-1,-5,7,2,16,7,18,9,0,19

15,7,-5,13,-3,3,-3,7,-4,11

parent read from the shared memory:

-1,-5,7,2,16,7,18,9,0,19

15,7,-5,13,-3,3,-3,7,-4,11

The two vectors are orthogonal? no

[kwang@computer][~/temp]$ ./A8p2\_unixpipe

child sent into the unnamed pipe:

-3,-1,4,-4,16,1,10,7,10,7

3,-2,-4,-1,-2,-2,-3,1,2,8

parent received from the unnamed pipe:

-3,-1,4,-4,16,1,10,7,10,7

3,-2,-4,-1,-2,-2,-3,1,2,8

The two vectors are orthogonal? yes

[kwang@computer][~/temp]$ ./A8p2\_namedpipe

child sent into the named pipe:

12,4,18,-1,1,0,17,-3,13,2

15,5,10,-2,-1,9,14,6,5,3

parent received from the named pipe:

12,4,18,-1,1,0,17,-3,13,2

15,5,10,-2,-1,9,14,6,5,3

The two vectors are orthogonal? no

1. (2 pts) You need to submit a single makefile to compile all of your four programs in problems 1 and 2. Use A8p1, A8p2\_shm, A8p2\_unixpipe and A8p2\_namedpipe as the output program file names in the makefile.
2. (25 pts) Determine the scheduling results of the following four processes P1, P2, P3, P4 using the shortest-remaining-time-first scheduling algorithm by **drawing a Gantt chart** for the scheduling results (break tie, if any, by first-come-first-served algorithm). What is the average waiting time? Note: to save time, your Gantt chart can take the following format: 0~4 P1; 4~8 P2; 8~12 P3; 12~16 P4; .... That is, you don’t have to draw it like that in our slides. For example, the Gantt chart on page 11 of module 10 can be specified as follows: 0~24 P1; 24~27 P2; 27~30 P3.

|  |  |  |
| --- | --- | --- |
| Process | Arrival Time | Burst Time |
| P1 | 0 | 7 |
| P2 | 2 | 4 |
| P3 | 4 | 8 |
| P4 | 6 | 5 |

1. (15 pts) Suppose we have a queue of three processes P1, P2, P3 with burst time 7, 5, 3 respectively and a scheduler uses the Round Robin algorithm to schedule these three processes with time quantum 4. **Draw a Gantt chart** for the scheduling results. What is the turnaround time of P1, P2, P3 respectively? The Gantt chart can be specified in a similar fashion as in the last question.

A total of six files (4 source files +1 makefile +1 solution file for questions 4 & 5) should be submitted.