Milestone #1 should satisfy requirements (2), (3), (4), (5), (6), (7), (8), and (9). You most probably need shared memory and semaphores for implementing the eight requirements.

(1) The main (parent) process creates a shared memory, which contains a variable, “counter” (in type :”int”). The content of “counter” should be initialized by ‘0’. (Milestone #2)

(2) The main (parent) process creates four child processes. The first child becomes the “Adder” (“Process A”), and the second, third, and fourth child becomes the first, second, and the third reader (R1, R2, and R3 respectively).

(3) Each of the five processes needs to wait (without using a “busy-loop” or “spin-wait”) until all the other processes are created. After all four child processes are created, they become active.

(4) When the main (parent) process sees that all four child processes are active, then it becomes the “Subtracter” (“Process S”).

(5) Each of Adder and Subtracter repeats their jobs (“jobs” are defined later) “NUM_REPEATS” times (“NUM_REPEATS should be declared as a label using “#define” at the beginning of your cpp source code file).

(6) The three readers (R1, R2 and R3) repeat until both Adder and Subtracter complete their jobs.

(7) When both the Adder and Subtracter complete their jobs, the three readers terminate.

(8) When the Subtracter completes its jobs, it proceeds to (9) below.

(9) The parent process waits (without using a “busy-loop” or “spin-wait”) until all the other four child processes (A, R1, R2, and R3) to terminate, if any of the four processes has not terminated yet.

(10) The parent process displays the content of “counter” in the shared memory. (Milestone #2)

(11) The parent process deletes the shared memory. (Milestone #2)

(12) The parent process terminate (the parent always the last one to leave). (Milestone #2)
The required program structure:

Actual implementation:

A sample binary executable (CS314_P2_demo_01.out) is posted to the CS314-002 course home. The timing of the process execution (for all the five processes) should be as shown by the sample.

Milestone #2: to complete this project #2 (due: November 28th). Milestone #2 for this project (Project #2) replaces “Programming Project #3” in this course. The requirements for Milestone #2 are as specified in the first Project #2 handout.