Summary on System V semaphore and POSIX semaphore

- One marked difference between the System V and POSIX semaphore implementations is that in System V you can control how much the semaphore count can be increased or decreased; whereas in POSIX, the semaphore count is increased and decreased by 1.
- POSIX semaphores do not allow manipulation of semaphore permissions, whereas System V semaphores allow you to change the permissions of semaphores to a subset of the original permission.
- Initialization and creation of semaphores is atomic (from the user's perspective) in POSIX semaphores.
- From a usage perspective, System V semaphores are clumsy, while POSIX semaphores are straight-forward
- The scalability of POSIX semaphores (using unnamed semaphores) is much higher than System V semaphores. In a user/client scenario, where each user creates her own instances of a server, it would be better to use POSIX semaphores.
- System V semaphores, when creating a semaphore object, creates an array of semaphores whereas POSIX semaphores create just one. Because of this feature, semaphore creation (memory footprint-wise) is costlier in System V semaphores when compared to POSIX semaphores.
- It has been said that POSIX semaphore performance is better than System V-based semaphores.
- POSIX semaphores provide a mechanism for process-wide semaphores rather than system-wide semaphores. So, if a developer forgets to close the semaphore, on process exit the semaphore is cleaned up. In simple terms, POSIX semaphores provide a mechanism for non-persistent semaphores.

- To compile POSIX semaphore based Producer Consumer problem

  gcc -o PC-posixSem buffer.h PC-posixSem.c –pthread

  Then execute the program

  ./PC-posixSem 5 2 2 (for an example)

- To compile fileio1.c

  gcc –o fileio1 fileio1.c

  Then execute the program

  ./fileio1 sourcefile copiedfile (Note: sourcefile needs to be a big file)

- .To compile fileio2.c

  gcc –o fileio2 fileio2.c

  Then execute the program

  ./fileio2 sourcefile copiedfile (Note: sourcefile needs to be a big file)

- To compile fileioS.c

  Then execute the program

  ./fileioS sourcefile copiedfile (Note: sourcefile needs to be a big file)

- To see the difference between sourcefile and copiedfile
  - diff sourcefile copiedfile