System V Posix

SYSTEM V	POSIX
SYSTEM V IPC covers all the IPC mechanisms viz., pipes, named pipes, message queues, signals, semaphores, and shared memory. It also covers socket and Unix Domain sockets.	Almost all the basic concepts are the same as System V. It only differs with the interface
Shared Memory Interface Calls shmget, shmat, shmdt, shmctl	Shared Memory Interface Calls shm_open, mmap, shm_unlink
Message Queue Interface Calls msgget, msgsnd, msgrcv, msgctl	Message Queue Interface Calls mq_open, mq_send, mq_receive, mq_unlink
Semaphore Interface Calls semget, semop, semctl	Semaphore Interface Calls Named Semaphores sem_open, sem_close, sem_unlink, sem_post, sem_wait, sem_trywait, sem_timedwait, sem_getvalue Unnamed or Memory based semaphores sem_init, sem_post, sem_wait, sem_getvalue,sem_destroy
Uses keys and identifiers to identify the IPC objects.	Uses names and file descriptors to identify IPC objects
NA	POSIX Message Queues can be monitored using select, poll and epoll APIs
NA	Multi-thread safe. Covers thread synchronization functions such as mutex locks, conditional variables, read-write locks, etc.
Requires system calls such as shmctl(), commands (ipcs, ipcrm) to perform status/control operations.	Shared memory objects can be examined and manipulated using system calls such as fstat(), fchmod()
The size of a System V shared memory segment is fixed at the time of creation (via shmget())	We can use ftruncate() to adjust the size of the underlying object, and then re-create the mapping using munmap() and mmap() (or the Linux-specific mremap())

Semaphore (System V):
Semaphore-Operationen blockieren wartende Prozesse in blocked Queues anstatt sich im Busy-Waiting-Modus zu befinden.

- Status des Prozesses wird auf Waiting gesetzt.
- Kontrolle geht an den CPU-Scheduler, der einem anderen Prozess die CPU zuteilen kann.

Quelle:
https://www.tutorialspoint.com/inter_process_communication/inter_process_communication_system_v_posix.htm