| **Basis for Comparison** | **Semaphore** | **Mutex** |
| --- | --- | --- |
| Basic | Signalling mechanism. It used to signal the other processes | It determines the locking of the shared resource and if any process locks the resource then no other process is permitted to access that particular resource for read/write operations until the process locking it unlocks it. |
| Value | Semaphore is an integer variable, Binary. | Mutex is an object. |
| Function | Semaphore allow multiple program threads to access a finite instance of resources simultaneously. | Mutex allow multiple program thread to access a single resource single instance but not simultaneously. |
| Ownership | Semaphore value can be changed by any process acquiring or releasing the resource. | Mutex object lock is released only by the process that has acquired the lock on it. |
| Types | Semaphore can be categorized into counting semaphore and binary semaphore. | Mutex is not categorized further. |
| Operation | Semaphore value is modified using wait() and signal() operation. | Mutex object is locked or unlocked by the process requesting or releasing the resource. |
| Resources Occupied | If all resources instances are being used, the process requesting for resource performs wait() operation and block itself till semaphore count satisfies the condition to enter the critical section | If a mutex object is already locked, the process requesting for resources waits and queued by the system till lock is released. |

Source Url: https://techdifferences.com/difference-between-semaphore-and-mutex.html