**Semaphores**

→ Integer variable, Binary

→ Used in mutual exclusion where multiple processes are stuck and execute in race around conditions

→ achieve process synchronization

Two types:

→ Counting: can keep values from 0 to infinity

→ Binary: 0 and 1 values only

Nomenclature:

P(), semwait(), down(), wait()→ Entry section

V(), semsignal(), up(), signal()→ Exit section

Three phases of mutual exclusion:

Entry of a process;

Critical Region;

Exit Critical region;

down(Semaphore s)→ Entry CS= free

{ s=3

s.value= s.value-1; P1: s=2, CS, s=0

if(s.value<0) P2: CS

{ P3: s=0 P4: s=-1

put the current process in suspended list: P2, P3, P4 Ready→ Blocked

}

else

{

let the process enter critical section; P1 → Execution

}

up(Semaphore s)→ Exit P1

{

s.value=s.value+1;

if(s.value>=0)

{

select a process from the wait list; sending the process into ready state;

wakeup(P2);

}

}

s=10

P()= 4 → Blocked

V()=2 → Exit

P()=5 → Waiting

10-4= s=6

6+2→ s=8

8-5 → s=3