Semáforo binario

```
struct SEMAPHORE {
    int valor {0,1};
    queue cola_de_bloqueo;
}
```

waitB(s):

```
if s.valor = 1
    s.valor = 0
else {
    poner el proceso en s.cola_de_bloqueo;
    bloquear este proceso;
    }
```

signalB(s):

```
if s.cola_de_bloqueo = vacía
    s.valor = 1
else {
    quitar un proceso p de s.cola_de_bloqueo;
    poner el proceso p en la cola de listos;
    }
```

Las primitivas **wait** y **signal** son **operaciones atómicas**

Semáforo generales v1

```
struct SEMAPHORE {
    int contador;
    queue cola_de_bloqueo;
}
```

waitB(s):

```
s.contador--;
if s.contador < 0 ;
{
  poner este proceso en s.cola_de_bloqueo;
  bloquear este proceso;
}
```

signalB(s):

```
s.contador++;
if s.contador <= 0
{
  quitar un proceso p de s.cola_de_bloqueo;
  poner el proceso p en la cola de listos;
}
```

Las primitivas **wait** y **signal** son **operaciones atómicas**

Semáforo generales v2

```
struct SEMAPHORE {
        uint contador;
        uint bloqueado:
        queue cola_de_bloqueo;
}
```

```
waitB(s):
        if s.contador = 0 {
          s.bloqueados++;
          poner este proceso en
          s.cola_de_bloqueo;
          bloquear este proceso;
        }
        else
          s.contador--
```

```
signalB(s):
        if s.bloquedos = 0
            s.contador++;
        else{
            quitar un proceso p de
            s.cola_de_bloqueo;
            poner el proceso p en la cola
           de listos;
           s.bloqueados--;
        }
```

Las primitivas **wait** y **signal** son **operaciones atómicas**