Critical Regions

- High level synchronization construct
- A shared variable of type T, is declared as:
  - var *v*: shared T;
- Variable *v* is accessed only inside statement:
  - **region** *v* **when** B **do** S;
- Where B is a boolean expression
- While statement S is being executed, no other process can gain access to variable *v*

Critical regions

- Regions to the same shared variable exclude each other in time
- When a process tries to execute the region statement, it gains access to shared variables, evaluates the boolean expression B, if true then proceed to statement S, else the process is delayed until no other process is accessing the shared region and B is true

Example, Bounded Buffer

- Shared variables
  - Var buffer: shared record
    - pool: array[0..n-1] of item;
    - count, in, out : integer;
- Producer process inserts nextp into shared buffer:
  - Region buffer when count < n do {
    - pool[in] := nextp;
    - in := in + 1 mod n; count := count + 1;
  - }

Example, Consumer

- Consumer process removes an item from the shared buffer and puts it in nextc:
  - Region buffer when count < 0 do {
    - nextc := pool[out];
    - out := out + 1 mod n; count := count - 1;
  - }

Ejemplos

- Agua
- El puente angosto
  - Capacidad máxima
  - Intercambio de sentido
- Misiones y caníbales
- Utilización de los baños
- Supersticioso
- Fumadores (Patil)

Ejemplos

- Robot
- Bar
- Santa Claus
- Blanca nieves