# Exclusión mutua para n-procesos

## 1. Acerca de exclusión mutua para N - Procesos

Se requiere 2 procesos en bucle infinito, donde los procesos no se pueden detener en sus secciones críticas, para ello los procesos no se pueden entrelazar unos con otros, siendo necesario protocolos antes y después de las secciones críticas.

Deben evitarse los interbloqueos, es decir, si varios procesos quieren acceder a la sección crítica, uno debe de conseguirlo, así mismo si dos hilos ejecutados a la vez por el mismo programa, acaban compartiendo un mismo recurso.

## 2. Algoritmo de Eisenberg-Mcguire

Reduce la espera de turnos mediante bucles infinitos, cumpliendo con:

- Satisfacer el requerimiento de la exclusión mutua.
- Satisfacer el requerimiento del progreso en ejecución.
- Cumplir con el requerimiento de espera limitada.

- **Repeat**

```
Flag(i) = Intento                          (quiere entrar)
j = turno
while j not = i  do
        if flag(j) not = Ocioso then j = turno
                else j = (j+1) Mod n
flag(i) = En-SC                            (en sección crítica)
j = 0
while (j < n) and (j=i or flag(j) not = En-SC) do j = j+1;
Until (j ≥ n) and (Turno = i or flag(Turno) = Ocioso);
        SECCION CRITICA
j = Mod n(Turno + 1)
While (j not = Turno) and (flag(j) = Ocioso) do j = (j+1) Mod n;
Turno = j
flag(i) = Ocioso
```

## 3. Estructura de datos

- Un indicador arreglo desde “0” hasta “n-1”, que contiene elementos de tipo enumerado, siendo los valores que indiquen:
    - Indica que el proceso no se encuentra ni desea entrar a la sección critica (se llamaría “restoproceso”).
    - Indica que el proceso desea entrar en su sección critica (se llamaría “quierentrar”).
    - Indica que el proceso se encuentra ejecutando su sección critica (se llamaría “enSC”).

- Un índice desde “0” hasta “n-1”: Indicara el momento en que un proceso puede entrar en la sección crítica.
- Por defecto todos los procesos estarán en un estado no activo (restoproceso) y el valor inicial del índice puede ser desde 0 hasta n-1.

# 4. Descripción del algoritmo

- Mediante un bucle, comprueba dentro rango de procesos, desde el que tenía permiso para entrar y el proceso actual, si los procesos estén en estado inactivo (restoproceso).
- Volverá a comprobar desde el principio si algún proceso “i” no está en el rango de procesos inactivos (resto proceso), además el proceso “i” entrara a su sección critica solamente si el indicador “j” es distinto en “enSC”.
- El valor de la variable índice puede ser modificado cuando un proceso entra a su sección crítica, y si no hay ningún proceso en su sección critica, el valor de índice permanece constante.
- Hasta que ningún proceso está en la sección critica dentro del índice “i”, el proceso actual tendrá activo su turno.
- Solo dará turno al siguiente que quiera entrar según la ordenación cíclica (índice 0 hasta índice n-1), si ninguno no quiere entrar, se queda con el turno que tiene.

```
    process P;
repeat
  repeat
      indicador[i]: =quiereentrar;
  (1) j: =indice;
      while(j≠i)
        begin
        if indicador[j]≠restoproceso
       (2) then j: =indice
           else j: =(j+1)mod n
        end;
  (3) indicador[i]: =enSC;
      j: =0;
      while ((j<n) and ((j=i) or (indicador[j]≠enSC)))
        j: =j+1;
  (4) until ((j>n) and ((indice=i) or
                          (indicador[indice]=restoproceso)));
      indice: =i;
      Sección Crítica;
      j: =(indice+1)mod n;
      while (indicador[j]=restoproceso)
        j: =(j+1)mod n;
  (5) indice: =j;
      indicador[i]: =restoproceso;
      Resto;
forever
```

## 5. Ejemplo

```
program EisenbergMcguire;
const N = 2;
var S : array[0..N] of integer;
        turno : integer;
        x : integer;

process type Proceso( i : integer );
var     j : integer;
begin
        repeat
                S[i] := 1;
                j := turno;
                while j <> i do
                begin
                        if S[j] <> 3 then
                                j := turno
                        else
                                j := (j + 1) mod N;
                end;
                S[i] := 2;
                j := 0;
                while (j < N) AND ((j = i) OR (S[j] <> 2)) do
                        j := j + 1;
        until ((j >= N) AND ((turno = i) OR (S[turno] = 3)));
        turno := i;
        x := x + i;
        writeln('P',i,' x = ',x);
        j := (turno + 1) mod N;
        while S[j] = 3 do
                j := (j + 1) mod N;
        turno := j;
        S[i] := 3;
end;

var aux : integer;
        proc : array[0..N] of Proceso;

begin
        for aux := 0 to N do
                S[aux] := 3;
        turno := random(N);
        x := 0;
        writeln('turno inicial = ',turno);
        cobegin
                for aux := 0 to N do
                        proc[aux](aux);
        coend;
end.
```