TC2037 – Implementación de Métodos Computacionales

Actividad 6.1 - Programación Concurrente en Erlang

1. Implementar el proceso **act6_1:suma()** que acepte un mensaje `{suma,N,P}`, donde `P` se asume que es un pid. Para cada mensaje, el proceso debe mandar `{respuesta, S}` a `P` (donde `S` es un acumulador).

   Programa de prueba.

```
prueba_suma() ->
   P = spawn(act6_1, suma, []),
   prueba_suma(5, P).
prueba_suma(N, P) when N > 0 ->
   P ! {suma, N, self()},
   receive
       {respuesta, S} ->
           io:format("Acumulado ~w~n", [S]),
           prueba_suma(N-1, P)
   end;
prueba_suma(_, _) ->
    io:format("Terminé mi trabajo~n").
```

   También tratar de probar el proceso **suma** interactivamente.

2. Implementar el proceso **act6_1:registro()** que maneje los siguientes mensajes:

   - `{registra, Nombre}` guarda `Nombre` en una lista si no está registrado.
   - `{elimina, Nombre}` elimina el `Nombre` de la lista si está registrado
   - `{busca, Nombre, P}` manda el mensaje `{encontrado, E}` al proceso `P`. `E` debe ser el átomo `si` o `no` dependiendo de si `Nombre` está registrado.
   - `{lista, P}` manda un mensaje de la forma `{registrados, L}` al proceso `P`. `L` debe ser la lista de nombres registrados.

Escribir la función **act6_1:prueba_registro()** como en el caso del problema anterior para probar la funcionalidad completa implementada en su código.