**TC2006 – Lenguajes de Programación***Programación Concurrente en Erlang*

Matrícula1\_\_\_\_\_\_\_\_\_\_ Matrícula2\_\_\_\_\_\_\_\_\_\_\_ Matrícula3\_\_\_\_\_\_\_\_\_\_\_

1. Implementar el proceso **conc:suma()** en Erlang 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(conc, suma, []),

prueba\_suma(5, P).

prueba\_suma(N, P) when N > 0 ->

P ! {suma, N, self()},

receive

{reply, S} ->

io:format("Acumulado ~w~n", [S]),

prueba\_suma(N-1, P)

end;

prueba\_suma(\_, \_) ->

io:format("Terminé mi trabajo~n").

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

1. Implementar el proceso **conc:registra** en Erlang que maneje los siguientes mensajes:

* {registra, Nombre} guarda Nombre en una lista si no 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.

Escribe la función **conc:prueba\_registra** como la función **prueba\_suma** del problema anterior para probar tu código.

1. Escribe el programa **conc:estrella(N, M, Mensaje)** que cree N procesos conectados en un estrella como el mostrado abajo. Una vez iniciados, estos procesos mandarán M veces el Mensaje alrededor de la estrella y terminarán con gracia cuando reciban un mensaje de terminar.

Cada proceso debe saber qué número de proceso es, cuantas veces ha recibido un mensaje y debe desplegar en una línea **“p: X – n: Y – m: Mensaje”** donde X es el # de proceso, Y es el número de veces que ha recibido el mensaje y Mensaje es el mensaje recibido.

