Práctica 03 - Computación Distribuida

Profesor: Salvador López Mendoza
Ayudante: Santiago Arroyo Lozano

30 de septiembre de 2024

Fecha de entrega

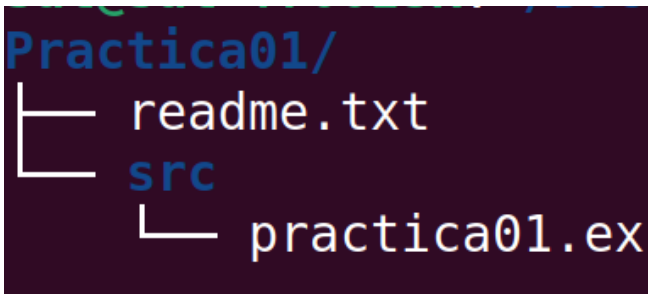
La práctica se entrega el **15 de octubre de 2024 a las 23:59**

Objetivo de la Práctica

Familiarizarse con el manejo de procesos en Elixir, como representar redes de procesadores en el lenguaje y problemas de elección de líder.

Lineamientos de entrega

Pueden incluir varios archivos .ex para desarrollar su práctica siempre y cuando haya un solo archivo principal que pueda correr las pruebas. Deberán acomodar sus archivos de la forma:



Donde el readme incluya anotaciones especiales de su práctica y el nombre y número de cuenta de los integrantes del equipo empezando por apellidos y dentro de `src` esté todo el código fuente.

Todas las funciones y módulos deberán ser documentadas debidamente usando el anotador `@doc` siguiendo los lineamientos del estándar para comentar código elixir. El módulo podrá llevar el nombre que prefieran (P01, Practica01, etc) sin embargo las firmas de las funciones deben permanecer exactamente como en las pruebas.

Se espera que verifiquen los tipos de sus funciones y no permitan que se hagan operaciones con parámetros del tipo incorrecto.

Parte 1 - Ejercicios sobre procesos

Incisos a resolver:

1. Escribe una función llamda **spawn_in_list/4** que spawnee n procesos de una función de un módulo particular y los almacene en una lista.

   Ejemplo de uso:

   ```
   iex(1)> Practica03.spawn_in_list(4, Grafica, :inicia, [])
   [#PID<0.117.0>, #PID<0.118.0>, #PID<0.119.0>, #PID<0.120.0>]
   ```

2. En ese contexto crea una función **genera/1** que spawnee n procesos de tu módulo Grafica de la práctica 02

   Ejemplo de uso:

   ```
   iex(1)> lista = Practica03.genera(4)
   [#PID<0.117.0>, #PID<0.118.0>, #PID<0.119.0>, #PID<0.120.0>]
   ```

3. Crea una función **send_msg/2** que le mande un mensaje particular a todos los procesos de una lista. (La lista es una lista de PIDs).

   Ejemplo de uso:

   ```
   iex(1)> Practica03.send_msg(lista, {:inicia})
   {:ok}
   ```

Parte 2 - Consenso

1. Implementa el algoritmo de consenso visto en clase a tu módulo Grafica de la práctica pasada. Tu implementación debe admitir los siguientes 3 mensajes:

    - `:proponer` - Este mensaje inicia a un nodo que propone un valor o una modificación a un recurso
    - `:consensuar` - Un nodo recibe este mensaje con un valor o recurso compartido y debe decidir si aceptarlo o no
    - `:comprobar` - Al final mandamos este mensaje para comprobar que todos los nodos aceptaron el mismo valor

Junto a este PDF se incluyen pocas pruebas unitarias para validar el consenso de un valor entero entre 3 nodos. Sin embargo se harán pruebas más avanzadas a la hora de revisión.

Si tienen problemas al importar su práctica a archivo de test tendrán que copiar y pegar los módulos todo en un solo archivo.

Mucho éxito, ya vamos a mitad de semestre, no desistan.