ALUMNOS:

Nicolas Natale: 108590

Santino Peiretti: 109320

ESTRUCTURAS UTILIZADAS

Se utilizaron distintas estructuras de datos (lista 1.0) asi como distintos tipos de structs y TDAs programados para un correcto funcionamiento de la red Algogram.

1. Tablas de Hash (ref 1.1)
2. Arboles Binarios de Busqueda (ref 1.2)
3. Colas de Prioridad de minimos (ref 1.3)
4. Pila (ref 1.4)

(lista 1.0)

COMANDOS

LOGIN:

El comando ***Login*** hace uso del ***struct Red***, el ***TDA Red*** y el ***TDA Pila***, el *TDA Pila* interno al *struct red* es usado para saber si hay o no un usuario logeado y mediante el *TDA Red* saber si tal usuario pertenece al registro de usuarios de la red.

LOGOUT:

El comando ***Logout*** hace uso de los mismos TDAs y struct antes mencionado para saber si hay o no un usuario logeado

PUBLICAR:

El comando ***publicar*** hace uso del *struct red* para saber si hay o no un usuario logeado y saber la cantidad de posts, para crear mediante el ***TDA Post***, un post del tipo *“id, nombre del posteador y el mensaje del post”*, luego se utiliza el *TDA Red* para guardar el post en una estructura *hash* de la forma *“clave: id; valor: post”* finalizando con el uso del ***TDA User*** para guardar el post en el feed de cada usuario el cual es un *heap* de minimos que se crea con cada usuario registrado de la red.

VER_SIGUIENTE_FEED:

El comando ***ver_siguiente_feed*** utiliza el *struct red* para saber si hay o no un usuario logeado, para luego mediante el *TDA User* saber si el feed del usuario contiene posts para ver, de serlo asi se desencola el post con mayor afinidad al usuario (ref 1.0) y haciendo uso del *TDA Post* para saber la informacion de post en cuestion.

LIKEAR_POST:

El comando *likear_post* hace uso del *struct red* para saber si hay o no un usuario logeado y el *TDA Red* para saber si, mediante el id pasado como parametro pertenece a los posteados, utilizando luego el *TDA Post* para guardar al usuario que realizo el like en una estructura ***ABB*** (requerimiento necesario para poder acceder a los likes del post en orden alfabetico).

MOSTRAR_LIKES:

El comando ***mostrar_likes*** no requiere de un usuario logeado para su funcionamiento, pero si requiere de un *id* pasado por parametro para poder hallar dicho post en el *TDA Red si es que pertenece y mediante el TDA Post y el TDA ABB* poder acceder a los likes del post en orden alfabetico.

AFINIDAD *(REF 1.0)*

El calculo de afinidad se determina como la diferencia de posiciones de los usuarios en el registro de usuarios de la red, si la diferencia de posiciones son iguales, se hace uso del id de los post para determinar el orden de salida.

Referencias.

[1.1] Hash Table: https://en.wikipedia.org/wiki/Hash_table

[1.2] ABB: https://en.wikipedia.org/wiki/Binary_search_tree

[1.3] Heap: https://en.wikipedia.org/wiki/Heap_(data_structure)

[1.4] Stack: https://en.wikipedia.org/wiki/Stack_%28abstract_data_type%29