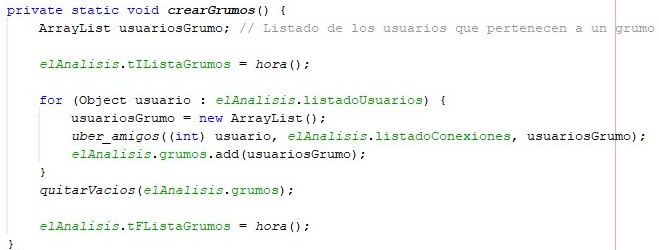
# **ANÁLISIS LISTA GRUMOS**

Una vez que se ha procesado el/los documento/s y, por tanto, se han procesado también la lista de usuarios y la lista de conexiones, es momento de determinar cuáles son los grumos de la red social.

Para crear los grumos serán necesario el objeto Análisis, un ArrayList en el que se guardarán los usuarios procesados y un ArrayList que almacenará los usuarios que ya han sido procesados.

Cuando se ha finalizado la ejecución de la función pedir datos, se ejecuta la función crearGrumos(), en la cual se recorre el ArrayList del listado de usuarios, se crea un nuevo ArrayList que será donde se almacenen los usuarios pertenecientes al grumo (en cada iteración se creará un nuevo ArrayList), se llamará a la función uber\_amigos() y se almacenará la lista de los usuarios del grumo en el ArrayList de grumos.

A continuación, una captura de la función crearGrumos():



La función uber\_amigos() recibe los siguientes parámetros:

* El usuario inicial, de tipo int, que será el usuario del que se quieran saber sus conexiones.
* El listado de conexiones existentes en el archivo (procesado anteriormente).
* El listado de los usuarios que pertenecen a un grumo.

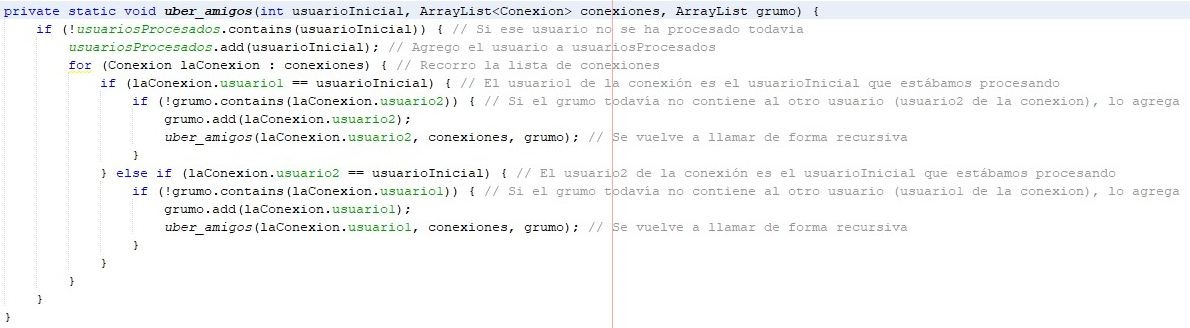
Lo primero que hace la función uber\_amigos() es comprobar si el usuario ya ha sido procesado con anterioridad o no. Si no ha sido procesado, continúa. Si sí que ha sido procesado, termina la ejecución de la función, volviendo a la función crearGrumos() y seleccionando el siguiente usuario.

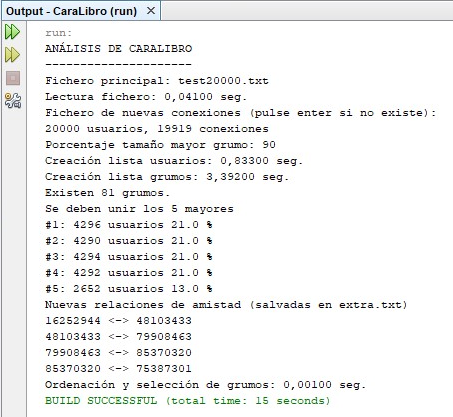
Después de comprobar si se ha procesado o no, si no ha sido procesado entonces añade el usuario al ArrayList de usuarios procesados. Luego, recorre la lista de conexiones en busca de ese usuario (ya sea siendo el primer usuario de la conexión o el segundo). En cualquiera de los dos casos, agregará tanto al primer usuario como al otro usuario de la conexión al grumo (si no está ya en la lista, realizando una comprobación para cada uno) y llamará de forma recursiva a la función uber\_amigos(), pero esta vez el usuario principal será el que no estaba en el grumo.

Cuando finaliza toda la ejecución de la función crearGrumos(), es decir, se ha recorrido toda la lista de usuarios de la red, se ejecuta la función quitarVacios(), que recibe por parámetro un ArrayList y elimina las posiciones del mismo que estén vacías. En este caso, el ArrayList que se pasa por parámetro es el de los grumos.

Funciona con un bucle do…while(), donde dentro recorre toda la lista en busca de que haya una sublista vacía. Si la encuentra, elimina dicha posición y continúa.

A continuación, la función uber\_amigos() y la salida por pantalla tras el proceso (utilizando el archivo de pruebas test20000.txt):





Como se puede observar, el programa procesa correctamente los grumos (sabe cuántos grumos hay, en este caso 81).

En el siguiente documento se analizará cómo se ordenan los grumos, cómo se agregan nuevas conexiones extra y resultados.