ANÁLISIS DE ORDENACIÓN Y SELECCIÓN DE GRUMOS

Una vez que se ha procesado el documento y se han creado de manera correcta los grumos, es paso de ordenarlos y seleccionar los necesarios para cumplir con el porcentaje que el usuario desea.

Previamente, durante la ejecución de la función pedirDatos(), se le pidió al usuario que introdujera el porcentaje de mayor tamaño de un grumo. En este caso, lo que se busca es realizar una serie de conexiones entre unos usuarios de los grumos más grandes para conseguir que exista un grumo que englobe al porcentaje de usuarios deseado.

Tras haber finalizado la función crearGrumos(), se ejecuta la función ordenarGrumos(). Esta función será la encargada de ordenar los grumos de mayor a menor número de usuarios. Para ello, recorre el listado de grumos y recoge el primero y el siguiente (es decir, durante la primera iteración cogerá el grumo 1 y el 2, en la segunda el 2 y el 3, en la tercera el 3 y el 4, y así sucesivamente), y va comparándolos hasta que ya no exista un grumo siguiente (final del listado). La comprobación que realiza es si el segundo grumo recogido es mayor en número de usuarios que el primero. Si así es, se almacena ese grumo en un ArrayList temporal (para evitar pérdida de datos), se elimina el grumo de la lista y se añade, con la función add(), el ArrayList temporal en la posición del anterior grumo.

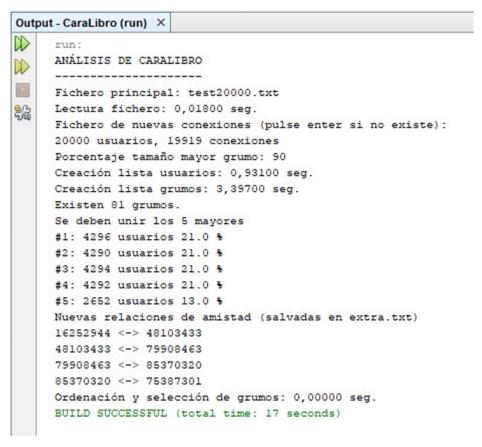
La función add() permite dos parámetros, el primero el índice en el que se quiere guardar un objeto, y el segundo el objeto. Si se agrega un objeto en una posición en la que ya existía otro, éste será desplazado a la posición siguiente. Por ello, al finalizar la ejecución, los grumos estarán ordenados de mayor a menor número de usuarios.

Finalizada la ordenación, queda seleccionarlos. Para ello, se recorre la lista de grumos (ya ordenada) y se van sumando los usuarios del grumo actual y el siguiente hasta que se alcance el porcentaje deseado por el usuario. El porcentaje se calcula en cada iteración para comprobar que sea correcto. Una vez que el porcentaje calculado es superior o igual al porcentaje deseado, se corta la ejecución del bucle y se imprime por pantalla los resultados (número de grumos mayores que han de unirse y los usuarios que se deben unir.

Los usuarios a unir siguen un orden de i, es decir, que si el primer grumo es el grumo 1 (i) y el segundo es el grumo 2 (i+1), será el usuario 1 del grumo 1 el que se una con el usuario 2 del grumo 2. Si en algún momento no existiera el usuario necesario (por ejemplo, si hay 5 usuarios y se necesita unir con el usuario 6), volvería a seleccionar el primero.

Finalmente, ejecuta la función salvarConexiones, que almacenará en el archivo "extra.txt", en cada línea, las conexiones necesarias para que, ejecutando por segunda vez el programa y seleccionando un archivo de conexiones extensas, se compruebe que las conexiones se realizan correctamente y que el grumo mayor es el deseado.

A continuación, captura del resultado del proceso, creando un grumo que englobe al 90% de los usuarios:



Output - CaraLibro (run) × run: ANÁLISIS DE CARALIBRO Fichero principal: test20000.txt Lectura fichero: 0,01500 seg. Fichero de nuevas conexiones (pulse enter si no existe): extra.txt 20000 usuarios, 19923 conexiones Porcentaje tamaño mayor grumo: 90 Creación lista usuarios: 1,01300 seg. Creación lista grumos: 3,74400 seg. Existen 77 grumos. El mayor grumo contiene 19824 usuarios (99.0)% No son necesarias nuevas relaciones de amistad. Ordenación y selección de grumos: 0,00000 seg. BUILD SUCCESSFUL (total time: 16 seconds)