Laboratorio N°1

Estructuras de datos y Algoritmos

Integrantes: - Cristian Rodríguez

-Jonás Oviedo Morales

Profesor: Martín Gutiérrez

Grupo 16

Sección: Sección 4

Introducción

Nuestro Código debe “Volcar” archivos ".csv” en un ArrayList de ArrayLists, para luego implementar Pilas (Stack) y colas(Queue). El código debe procesar los datos referente a los archivos “.csv”(dentro de la carpeta “csv”, la cual adjuntamos también, para evitar tener problemas en la compilación del código), y procesar los datos para crear archivos de salida, para eso usaremos tripletas (objetos que almacenan 3 datos), que tendrán la información del archivo de entrada y nos permitirán ordenarlas con el método “.sort”, luego introducir las tripletas ya ordenadas a una Cola para luego cambiar el orden con una pila.

Explicacion del codigo

Class Cola y Pila:

Tanto como en la pila como en la cola usamos la implementación del libro sin mucho cambio extra.

Class Tripleta

3-7: Crea la clase “Tripleta que “hereda” métodos de “Comparable” y define a un String de categoría, uno de “NombreProducto” y un int para conteo.

9-22: Define 2 constructores, uno que no pide parámetros que instancia las variables conteo, Nombreproducto y categoría y otro que pide 2 parámetros String para asignarlos a el NombreProducto y a categoría, además instancia a conteo como 1.

23-41: Crea las funciones “Geters” y “Seters” de cada parametro.

43-51: Crea 2 funciones referentes a Conteo, la primera “IncConteo” incrementa el parametro “conteo” en 1, y “getConteo” retorna el valor de “conteo”.

53-59: Instancia el método “compareTo” de “Compare” y lo redefine con “@Override”, el método pide como parámetro un objeto, para luego hacer un casting de “Tripleta” en el y obtener el conteo de cuantas veces fue comprado ese objeto lo guarda en una variable “t2count” luego retorna la resta entre t2count y la tripleta que esta pidiendo la comparación.

Class Laboratorio 1:

10-14: Se define los SubStrings

15-35: Se define las variables File para recorrer los archivos en la carpeta “csv” y agregar los nombres de estos a un archivo “filenames”.

36-37: Se inicia un “try” y se recorre todos los archivos de la carpeta “cvs” por el nombre del archivo

36-47: Se crea el dataset que funciona como un “mapa” para almacenar en cada casilla un arreglo de la información de los archivos y se definen los archivos de salida la información del dataset ya trabajada.

48-64: Recorre un bucle hasta que el texto se acaba y divide los archivos por el marcador “|” y añade cada división a un arreglo para luego añadirlo al dataset

66-69: Extrae el nombre del archivo eliminando el “.csv” y “mx”.

71-95: Recorre el dataset buscando “datos” iguales en el caso de encontrarlos los aumenta el contador de dicho dato en la tripleta y si no lo encuentra crea la tripleta.

99: Ordena la tripleta

98-113: Mete los datos ordenados a la tripleta y los ordena desde el más comprado al menos comprado, para luego escribirlos en un archivo de salida.

101-135: Tomas los datos de la cola y los metes en una pila, luego los usas para escribir otro archivo de salida, con los datos invertidos.

137-149: Cierra los archivos de salida.

Opinión y análisis de la eficacia en la resolución de este laboratorio:

Dentro de todo el programa cumple su función sin complicarse tanto, pero hay un par de cosas dentro de la ejecución del programa que viéndolo desde una perspectiva más crítica, hay un par de cosas que a pesar que no dañan el objetivo de la actividad, pueden provocar ciertos errores. Por ejemplo, los .csv están en una carpeta separada, y el programa lee todos los archivos de esa carpeta, como en este caso, no es necesario compilar el programa dos veces no hay problema, pero si fuera el caso, si podrían suceder errores.

Dentro del programa en sí, el que consume mayor cantidad de tiempo es el de la creación de las tripletas con la comparación de nombres, ya que es un proceso que se repite tantas veces pienso que es el principal a tomar en cuenta si quisiera optimizar el programa. Los otros ciclos, en general, siento que hacen su trabajo eficientemente.

Fuentes de las que saqué ciertos códigos que adapté a mi programa:

https://stackoverflow.com/questions/1844688/how-to-read-all-files-in-a-folder-from-java