**UNIVERSIDAD CATÓLICA DE CÓRDOBA**

Ingeniería en Sistemas

**PARCIAL 2:**

**“**Análisis estadístico COVID-19 – Argentina”

MATERIA: *Programación III.*

PROFESOR: *Matías Sebastián Murgui.*

ALUMNOS: *Andrés Cabana, Santiago Agüero, Julián Gergolet.*

4 de noviembre de 2021

**INTRODUCCIÓN**

Debido al caso de pandemia del año 2020, quisimos hacer un visualizador de datos de los casos en Argentina. Para esto se quiere poder mostrar datos ordenados y agrupados en distintas categorías. Como así también hacer un procesado simple de los datos y mostrar alguna información extra.

**DESARROLLO**

Primero comenzamos defendiendo Casos.h, una clase que procesa los datos y los clasifica. Dicha clase la usaremos para resolver todos los argumentos posteriores. Decidimos esta estructura para que cuando tengamos que leer los datos, sean más transparentes y más fáciles de extraer.

Argumentos:

* ***Argumento -estad***:

Para el siguiente argumento utilizamos la función PrintStats, donde nos pareció conveniente la NO utilización de estructuras de datos para simplemente con una sola iteración (un while) mostrar los datos del archivo.

PrintStats se basa en ir línea por línea (con un getline) que finalizara (cortando el while) si falla o si no hay líneas.

* ***Argumento p\_casos[n] y p\_muertes[n]:***

Con estos caso decidimos usar la misma función porque ambos tienen que ordenar provincias. Nuestra función principal es PrintIncident, que recibe el archivo, un limite y un bool que determina si estamos analizando p\_casos o de p\_muertes.

Utilizamos dos clases llamadas Casos.h y Provincia.h. Esta ultima almacena el nombre de la provincia y un contador. Este contador lo utilizaremos a nuestra conveniencia según estemos hablando de muertes o casos.

Antes de terminar PrintIncident hacemos un QuickSort para acomodar las provincias de mayor a menor según el valor del contador.

* ***Argumento casos\_edadaños***:

Para este argumento usamos un HasMap y una Lista. Incluso logramos que corra el archivo de 4gb en un tiempo bastante considerable.

* ***Argumento casos\_cui[fecha]:***

Para resolver este argumento creamos una función que dentro contiene una lista. Nos pareció conveniente ya que utilizando el método insertar\_primero, para cargar los casos en la misma, es más rápido y conservamos el orden.

Al final hacemos un QuickSort de un arreglo unidireccional de casos, que los ordena según la fecha.

COMPLICACIONES:

Nos costó mucho comparar fechas, no recordábamos como hacerlo. Tuvimos que recurrir a internet y usar string comparison (strcmp).

El resto del código es bastante eficaz y eficiente, tomando un tiempo de ejecución muy chico cuando el número de casos no es extremadamente grande. En nuestras computadoras el archivo de 4gb no nos corre con fluidez y muchas veces se nos cierra el programa cuando ejecutamos ciertos argumentos, como casos\_cui o casos\_edadaños.

Todas las pruebas fueron realizadas en el archivo de cien mil (100.000) casos y en menor medida los de 10 y 1000 casos.