Enrique Mondelli - A01379363
Diego Mejía - A01024228

Reporte Actividad 2.3

Para el segundo trabajo en equipo, se utilizó el parser realizado en la entrega anterior que procesa la información por fila del archivo CSV, además de que se utilizó el ADT de la clase Registro para guardar todos los datos de cada registro. Con todo lo anterior que ha sido útil para el desarrollo de esta actividad, es posible establecer todo lo nuevo que se tuvo que implementar para la segunda actividad.

Para poder hacer respuesta a las preguntas previamente definidas, se estableció una clase que tuviera los siguientes atributos privados:

- *IP* de tipo string que sirve para la construcción de la dirección de IP completa que será buscada en el archivo CSV.
- *name* de tipo string que es el nombre de la computadora con su IP correspondiente.
- *conexiones_entrantes* de tipo stack<Registro<string>> que es donde se guardan las conexiones entrantes. Se decidió hacer uso de la estructura *stack* porque permite leer su información desde la última posición hasta la primera, el primer valor ingresado estando en la parte inferior de la pila y el último valor ingresado en la parte superior de la pila.
- *conexiones_salientes* de tipo queue<Registro<string>> que es donde se guardan las conexiones salientes. Se decidió hacer uso de la estructura *queue* porque permite leer su información desde la primera posición hasta la última, el primer valor ingresado estando en el frente de la fila y el último valor ingresado estando hasta el final de la fila.
- *registros de tipo vector<Registro<string>>* que es donde se guardan todos los registros, sin importar el tipo de valor gracias a la implementación del ADT, y que es también el lugar de donde se sacan las conexiones entrantes y salientes.

Además de esto, se realizaron los siguientes métodos públicos:

- *get_IP* para obtener el nombre completo de la dirección IP.
- *print_conexiones_entrantes* para la impresión a terminal de una copia de la pila (para no eliminar elementos de la pila original) conexiones entrantes desde la cima de la pila hasta

la parte inferior y eliminando cada conexión siguiendo ese proceso, utilizado para comprobar que la estructura elegida fuera la correcta para estas conexiones.

- *print_conexiones_salientes* para la impresión a terminal de una copia de la fila (para no eliminar elementos de la fila original) conexiones salientes desde la parte derecha de la fila hasta la izquierda y eliminando cada conexión siguiendo ese mismo proceso, utilizado para comprobar que la estructura elegida fuera la correcta para estas conexiones.
- *ultima_conexion_entrante* para obtener la conexión en la cima de la pila, o no regresar nada si la pila está vacía.
- *get_ce_size* para obtener el tamaño de la pila.
- *get_cs_size* para obtener el tamaño de la fila.
- *get_registros* para obtener el vector de registros.

Ambas de estas estructuras, tanto la pila cómo la fila, tienen que ser recorridas de manera lineal para su impresión, así que su tiempo de complejidad para ambas sería O(n). Y para obtener el frente o la cima de las estructuras, O(1) gracias a la forma en que los elementos están ordenados.