



Taller 02 Estructuras lineales

Juan Sebastián Rodríguez Pabon

David Esteban Rodríguez Jurado

Daniel Galvis Jaramillo

Julián Pérez

Dirigido a John Corredor

23 de septiembre de 2024

Bogotá D.C

Introducción

El objetivo es desarrollar un programa eficiente para la lectura y manipulación de archivos de texto, utilizando estructuras de datos de C++ para facilitar el manejo de palabras y la búsqueda de subcadenas dentro de un archivo. La estructura elegida, abordada a continuación, permite organizar las palabras del archivo según la línea en la que aparecen, optimizando las búsquedas específicas y mejorando la modularidad del código.

Este diseño asegura la escalabilidad y mantenibilidad del programa, con un enfoque en la eficiencia y la simplicidad.

Implementación

La implementación del programa está diseñada para facilitar la lectura y manipulación de un archivo de texto de manera estructurada. Se decidió utilizar un **vector<std::list<Palabra>>** para almacenar cada línea del archivo, dividiendo el contenido en palabras y asociándolas con su número de línea. Esto permite que las búsquedas de palabras sean más específicas y eficientes, ya que cada palabra se encuentra organizada según la línea en la que aparece.

La clase **Palabra** encapsula tanto la palabra en sí como el número de línea en el que se encuentra, lo que le otorga una mayor modularidad al manejo del texto. La clase **ArchivoTexto** se encarga de todas las operaciones relacionadas con las palabras del archivo, como las búsquedas, garantizando que el manejo del texto esté separado de la lógica principal del programa.

Para la separación de responsabilidades, el **main()** se limita a la lógica de ejecución principal, mientras que las operaciones complejas, como las búsquedas, están encapsuladas en funciones

dentro de las clases. Esto favorece la mantenibilidad del código, que es legible y fácil de modificar si es necesario.

Guía de compilación y ejecución

Para compilar el programa usando g++, sigue estos pasos:

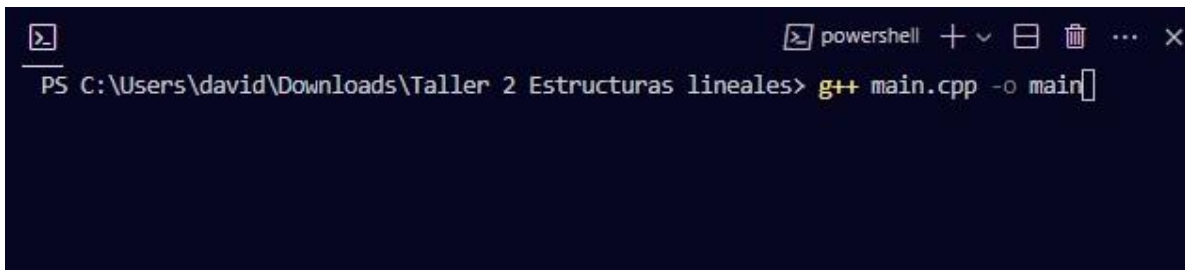
Compilar el código: Abre una terminal y ejecuta el siguiente comando para compilar el archivo **main.cpp** junto con el archivo **Clases.h** “**g++ main.cpp -o main**”.

Ejecutar el programa: Una vez compilado, puedes ejecutar el programa pasándole el archivo de texto como parámetro “**./main archivo.txt**”.

Asegúrate de que **archivo.txt** esté en el mismo directorio que el ejecutable o proporciona la ruta completa al archivo.

Compilación y ejecución

Para la compilación con g++ utilizamos el comando “**g++ main.cpp -o main**” y esperamos a que se compile el programa.



```
PS C:\Users\david\Downloads\Taller 2 Estructuras lineales> g++ main.cpp -o main
```

Figura 1. Comando para compilación.

Para comprobar que si se compiló y creó el ejecutable revisaremos el directorio y buscaremos el ejecutable con el nombre que creamos.

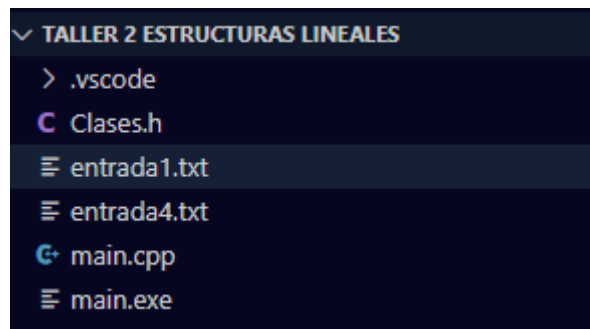


Figura 2. Buscar ejecutable.

Para ejecutar nuestro programa escribiremos el comando “**./main entrada1.txt**”, ya que

haremos la prueba con entrada1.txt que está en nuestro directorio, recordando que la estructura del archive que se da como parámetro tiene que tener como primera línea el número de líneas del archive de texto sin contarse a ella misma y la segunda línea es la subcadena con la que se hará el proceso de búsqueda de palabras en el archivo de texto.



```
entrada1.txt
1 5
2 co
3 solo con porro pongo loco
4 loco monto moto, corro, todo borro
5 borro todo tonto, no compongo, solo compro
6 yo, mono loco, no conozco otro modo!
```

Figura 3. Captura del archivo

“entrada1.txt”.

```
PS C:\Users\david\Downloads\Taller 2 Estructuras lineales> ./main entrada1.txt

Hay 3 palabras que empiecen con: co
línea 4: corro,
línea 5: compongo,
línea 5: compro

Hay 5 palabras que contienen: co
línea 3: loco
línea 4: loco
línea 4: corro,
línea 5: compongo,
línea 5: compro

Hay 2 palabras que contienen: oc
línea 3: loco
línea 4: loco
```

Figura 4. Captura de la ejecución del

programa.

Si hacemos este proceso a mano veremos que nos dará exactamente el mismo resultado, siempre y cuando pasemos como parámetro desde la terminal un archivo de texto válido para el programa.

Conclusión:

El programa desarrollado demuestra una implementación eficiente y estructurada para la manipulación de archivos de texto en C++. La organización de las palabras por líneas mediante un vector<std::list<Palabra>> permite optimizar las búsquedas y facilita la expansión futura del sistema. Las operaciones específicas de manipulación del texto favorecen la claridad y el mantenimiento del código, haciéndolo más fácil de adaptar a nuevas necesidades. Con una correcta compilación y ejecución del programa, se ha comprobado su funcionalidad y precisión en las búsquedas de subcadenas dentro de archivos de texto, validando así el diseño modular del sistema implementado.