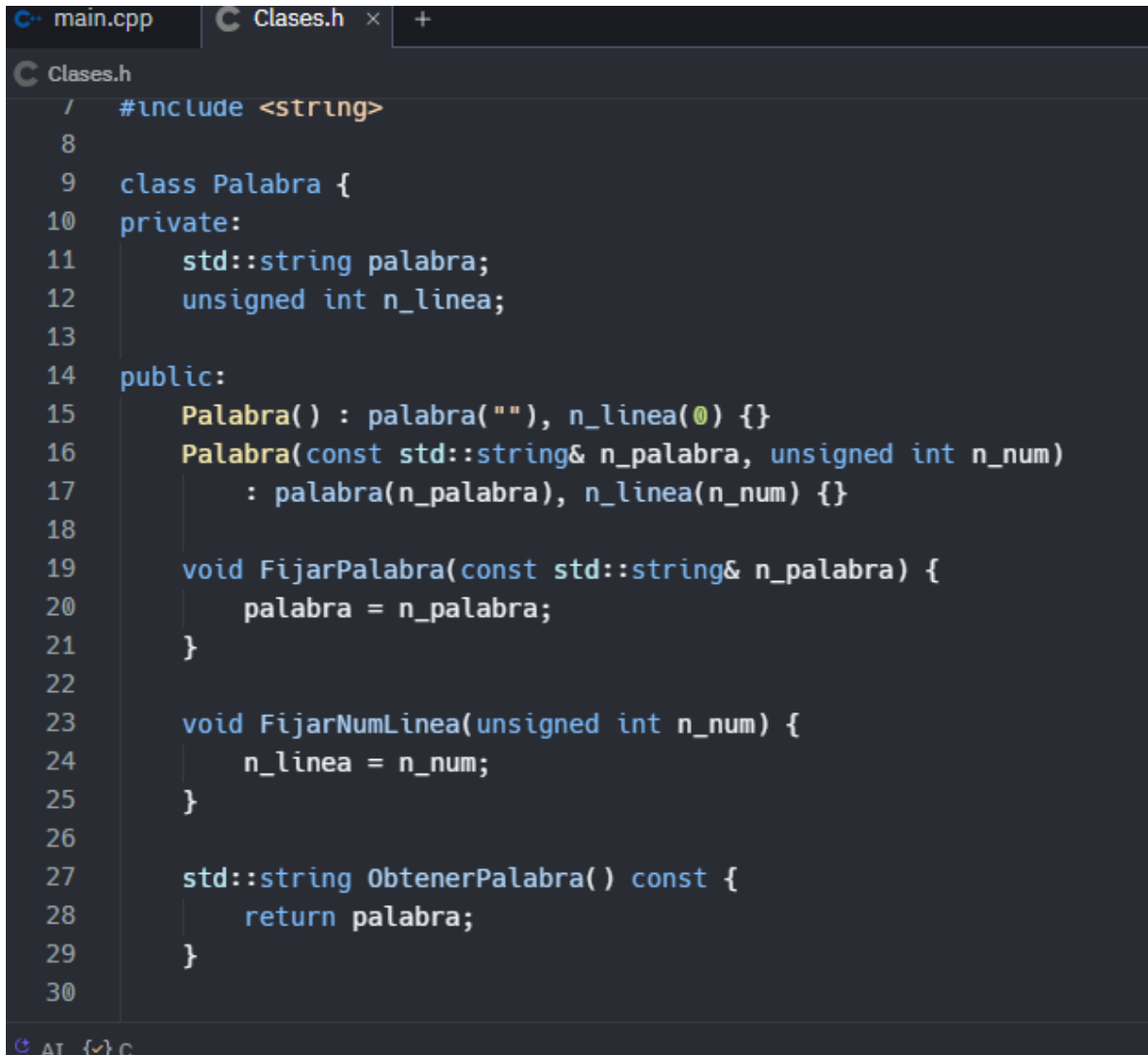


Informe del Código

Archivo Clases.h

Definición de la Clase Palabra



```
1 / #include <string>
2
3 8
4 9 class Palabra {
5 10 private:
6 11     std::string palabra;
7 12     unsigned int n_linea;
8 13
9 14 public:
10 15     Palabra() : palabra(""), n_linea(0) {}
11 16     Palabra(const std::string& n_palabra, unsigned int n_num)
12 17         : palabra(n_palabra), n_linea(n_num) {}
13 18
14 19     void FijarPalabra(const std::string& n_palabra) {
15 20         palabra = n_palabra;
16 21     }
17 22
18 23     void FijarNumLinea(unsigned int n_num) {
19 24         n_linea = n_num;
20 25     }
21 26
22 27     std::string ObtenerPalabra() const {
23 28         return palabra;
24 29     }
25 30
```

Descripción:

Palabra: Representa una palabra en el texto, asociada a una línea específica.

Datos:

palabra: Cadena de caracteres que almacena la palabra.

n_linea: Número de línea en la que se encuentra la palabra.

Operaciones:

- FijarPalabra: Cambia la palabra actual.
- FijarNumLinea: Cambia el número de línea actual.
- ObtenerPalabra: Retorna la palabra almacenada.
- ObtenerNumLinea: Retorna el número de línea actual de la palabra.

```
class ArchivoTexto {
private:
    std::vector<std::list<Palabra>> lineasTexto;
public:
    ArchivoTexto() {}

    void FijarListaLineas(const std::vector<std::list<Palabra>>& n_lista) {
        lineasTexto = n_lista;
    }

    std::vector<std::list<Palabra>> ObtenerListaLineas() const {
        return lineasTexto;
    }

    size_t ObtenerNumLineas() const {
        return lineasTexto.size();
    }

    void AgregarListaPals(const std::list<Palabra>& n_lista) {
        lineasTexto.push_back(n_lista);
    }
}
```

Descripción:

ArchivoTexto: Representa un archivo de texto como un vector de listas de Palabra, donde cada lista corresponde a una línea.

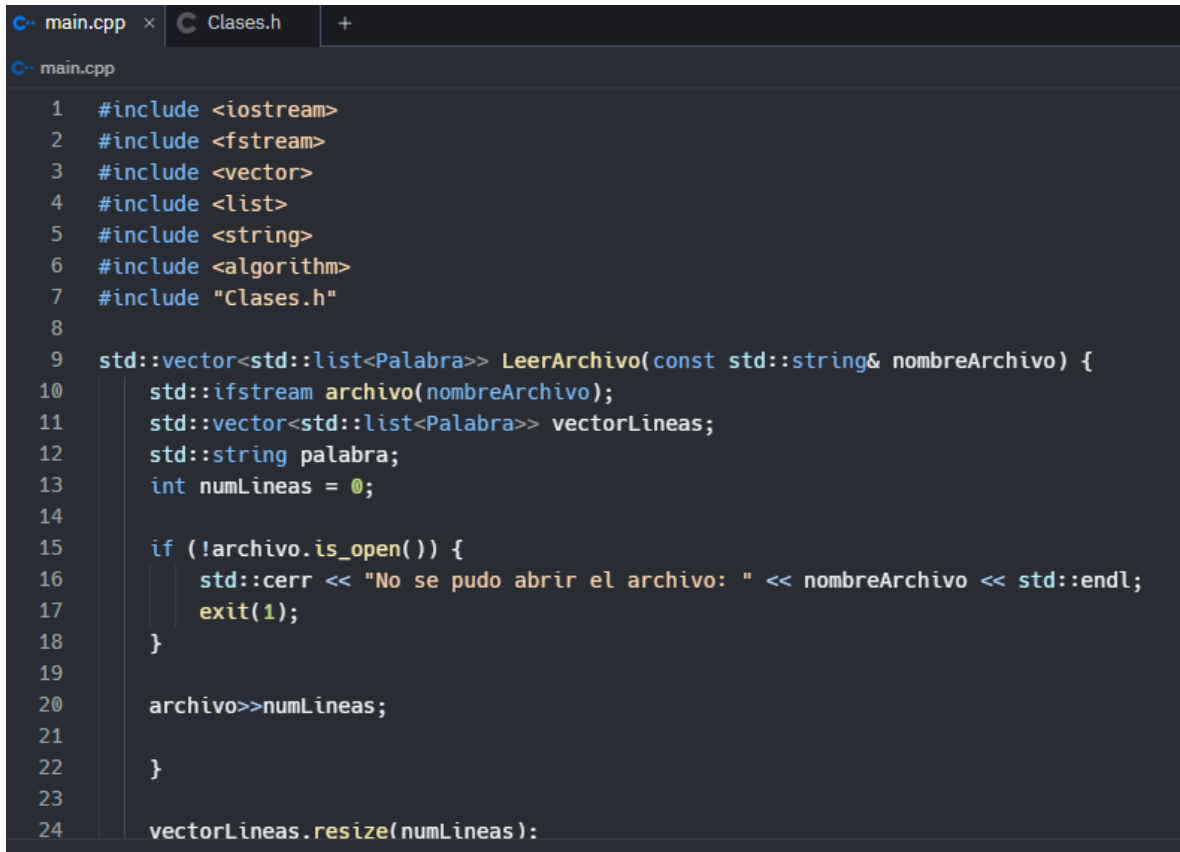
Datos:

lineasTexto: Vector de listas de Palabra, representando el archivo de texto.

Operaciones:

- FijarListaLineas: Establece la lista de líneas de texto.
- ObtenerListaLineas: Retorna el vector de líneas de texto.
- ObtenerNumLineas: Retorna el número de líneas de texto en el archivo.
- AgregarListaPals: Agrega una nueva línea de palabras al vector.

- BuscarPrincipio: Busca palabras que comiencen con una subcadena dada.
- BuscarContiene: Busca palabras que contengan una subcadena en cualquier posición.



```
1  #include <iostream>
2  #include <fstream>
3  #include <vector>
4  #include <list>
5  #include <string>
6  #include <algorithm>
7  #include "Clases.h"
8
9  std::vector<std::list<Palabra>> LeerArchivo(const std::string& nombreArchivo) {
10     std::ifstream archivo(nombreArchivo);
11     std::vector<std::list<Palabra>> vectorLineas;
12     std::string palabra;
13     int numLineas = 0;
14
15     if (!archivo.is_open()) {
16         std::cerr << "No se pudo abrir el archivo: " << nombreArchivo << std::endl;
17         exit(1);
18     }
19
20     archivo>>numLineas;
21
22 }
23
24 vectorLineas.resize(numLineas):
```

```

std::string InvertirCadena(const std::string& cadena) {
    std::string invertida = cadena;
    std::reverse(invertida.begin(), invertida.end());
    return invertida;
}

int main(int argc, char* argv[]) {
    if (argc < 2) {
        exit(1);
    }

    std::string nombreArchivo = argv[1];
    std::vector<std::list<Palabra>> vectorLineas = LeerArchivo(nombreArchivo);
    ArchivoTexto archivoTexto;
    archivoTexto.FijarListaLineas(vectorLineas);

    std::string subcadena;
    std::cout << "Ingresa una subcadena: ";
    std::cin >> subcadena;

    std::list<Palabra> palabrasInicio = archivoTexto.BuscarPrincipio(subcadena);
    std::list<Palabra> palabrasContiene = archivoTexto.BuscarContiene(subcadena);
    std::string subcadenaInvertida = InvertirCadena(subcadena);

```

Descripción:

Función LeerArchivo:

- Lee el archivo especificado y construye un vector de listas de Palabra. Cada lista representa una línea del archivo.

Función InvertirCadena:

- Invierte una cadena de caracteres.

Main:

- Abre el archivo y lee su contenido en un vector de listas de Palabra.
- Crea una instancia de ArchivoTexto y establece la lista de líneas.
- Solicita al usuario una subcadena para buscar palabras que comiencen con la subcadena, contengan la subcadena, y contengan la subcadena invertida.
- Imprime los resultados de las búsquedas, mostrando las líneas y las palabras encontradas.

Operaciones Principales:

- LeerArchivo: Procesa el archivo y convierte el texto en un formato estructurado.
- InvertirCadena: Utiliza la función estándar `std::reverse` para invertir la cadena.
- main: Coordina el proceso, maneja la entrada del usuario y muestra los resultados de las búsquedas.

Para compilar y ejecutar el código en una terminal Linux, sigue estos pasos:

Guarda los Archivos

Asegúrate de tener los archivos `Clases.h` y `main.cpp` en el mismo directorio.

Compila el Código

Usa el compilador `g++` para compilar el archivo `main.cpp`. Abre una terminal y ejecuta el siguiente comando:

```
g++ main.cpp -o programa
```

Esto generará un ejecutable llamado `programa`.

Ejecuta el Programa

Ejecuta el programa pasando el nombre del archivo de texto como argumento. Asegúrate de que el archivo de texto esté en el mismo directorio o proporciona la ruta completa. Por ejemplo:

```
./programa nombre_archivo.txt
```

Luego, ingresa la subcadena cuando el programa lo solicite.