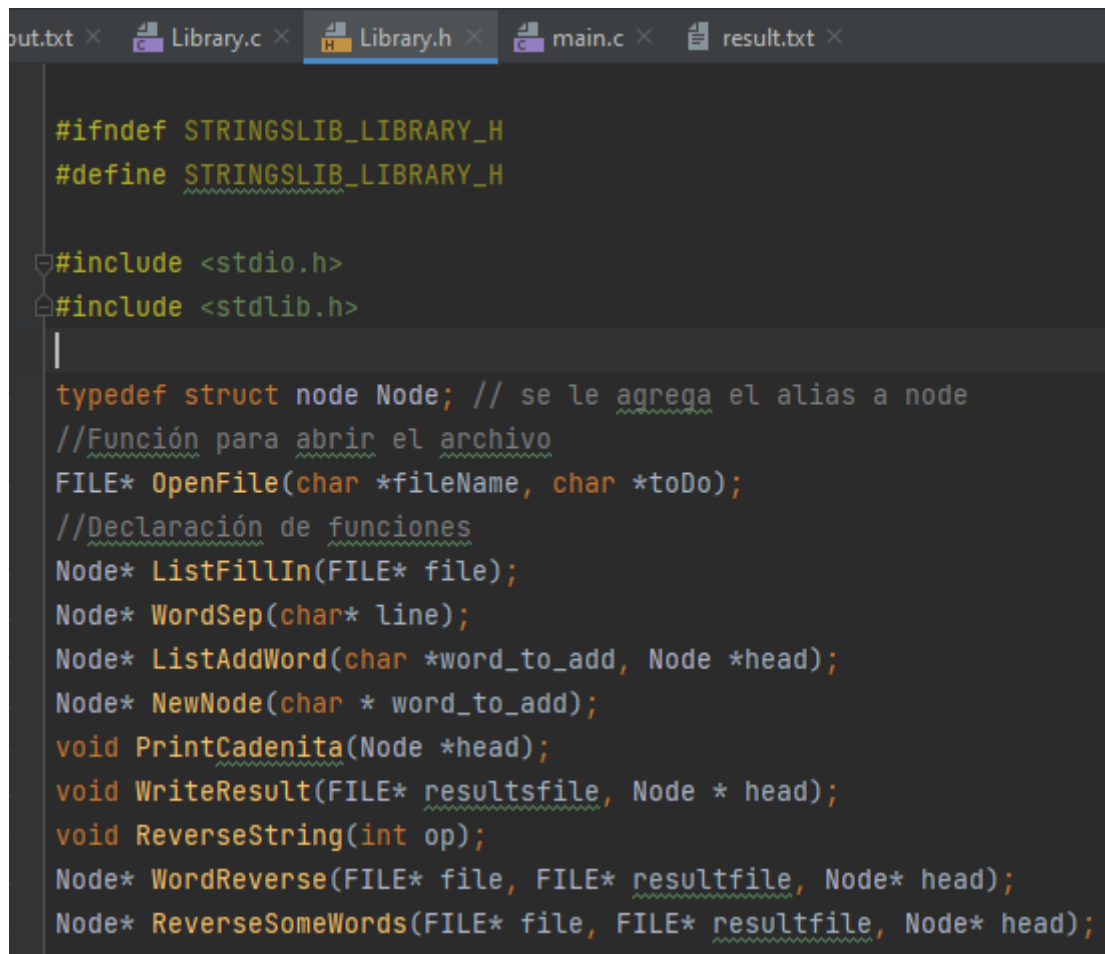


YAMILE: A comparación de las tareas previamente asignadas, mi equipo y yo decidimos realizar este trabajo desde casi el día que nos lo dejó, para así, no saturarnos ni estar con prisas, ya que nos quedó de aprendizaje no realizar códigos desde 1 o 2 días antes, la verdad es que esto nos sirvió mucho ya que dividimos tiempos súper bien, y alcanzamos a terminar el código a buena hora.

El programa esta implementado en el IDE de CLion, lenguaje C y cuenta con librería creada por nosotros

Tiene el nombre de “Library”, donde se almacenan las funciones



```
#ifndef STRINGSLIB_LIBRARY_H
#define STRINGSLIB_LIBRARY_H

#include <stdio.h>
#include <stdlib.h>

typedef struct node Node; // se le agrega el alias a node
//Función para abrir el archivo
FILE* OpenFile(char *fileName, char *toDo);
//Declaración de funciones
Node* ListFillIn(FILE* file);
Node* WordSep(char* line);
Node* ListAddWord(char *word_to_add, Node *head);
Node* NewNode(char * word_to_add);
void PrintCadenita(Node *head);
void WriteResult(FILE* resultsfile, Node * head);
void ReverseString(int op);
Node* WordReverse(FILE* file, FILE* resultfile, Node* head);
Node* ReverseSomeWords(FILE* file, FILE* resultfile, Node* head);
```

Nos funcionó como equipo tener al menos 2 versiones de intento de código de algunas funciones para así poder ver cual funcionaba de manera más eficiente o cual era la mejor opción, decidimos dejar la versión mas resumida en nuestro código final con un total de 190 lineas en Library.C, al inicio teníamos casi 300 pero lo simplificamos bastante.

El código cuenta con dos archivos .txt el input donde el usuario escribe las palabras, ya sea con espacio, salto de línea, una sola palabra etc. Y el result.txt le da al usuario el final esperado, por ejemplo, utilizando el ejemplo del profe:

```
input.txt x Library.c x Library.h x main.c x
1 Hola mundo esta es la primer linea
2 Esta linea tambien existe en el archivo
3 AAAA BCDE FGHIjk
4
```

Dentro de la función de reverse_string ponemos la opción que queramos ejecutar

```
t.txt x Library.c x Library.h x main.c x result.txt x
#include <stdio.h>
#include "Library.h"
#include <string.h>

int main() {
    // si se pasa 1 hará WordsReverse()
    // si se pasa 2 hará ReverseSomeWords()
    ReverseString(op: 2); // aqui se pone la opción

    return 0;
}
```

Resultado Op 1:

```
input.txt x Library.c x Library.h x main.c x
linea primer la es esta mundo Hola
archivo el en existe tambien linea Esta
FGHIjk BCDE AAAA
```

Resultado Op2: (ojo, solo tomando en cuenta los nodos mayor o igual a 5 caracteres)

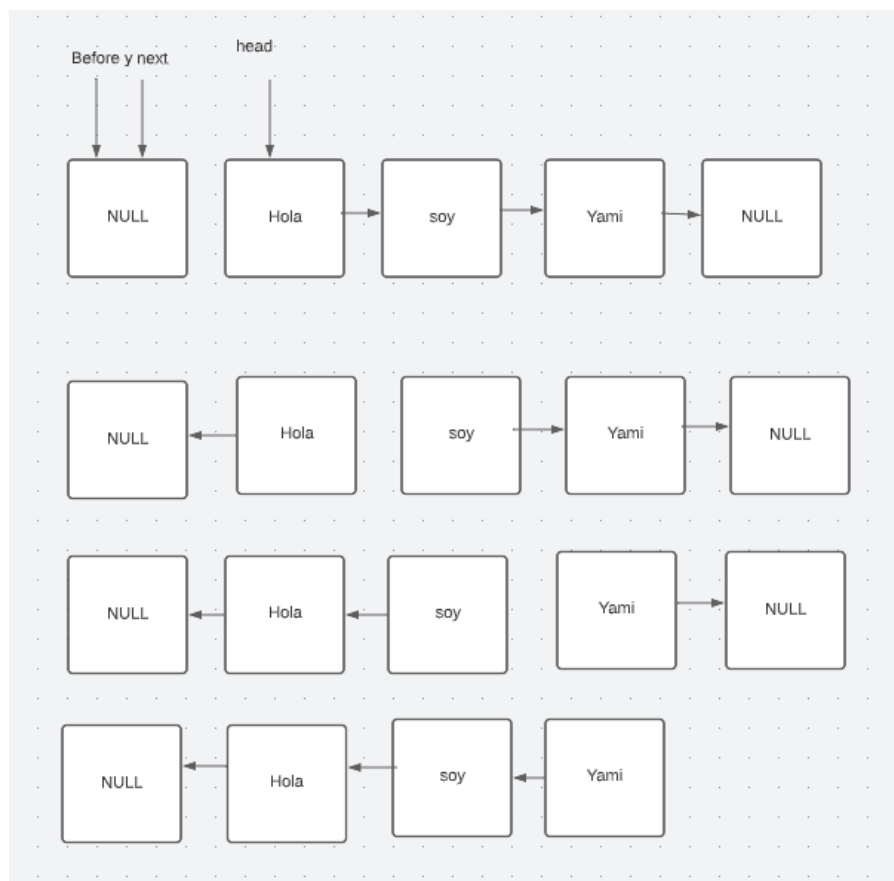
```
input.txt x Library.c x Library.h x main.c x
Hola odnum esta es la remirp aenil
Esta aenil neibmat etsixe en el ovihcra
AAAA BCDE kjIHGF
```

Tuvimos unos pequeños errores de “código limpio o sintaxis” que ya al final corregimos como por ejemplo quitamos unos do-while y los cambiamos por while, cosas así. También corregimos que los nombres de las funciones tuvieran un seguimiento de asignación específica para todos, por ejemplo, usamos Mayúsculas y que estuvieran sin espacio, porque teníamos unos con espacio o sin mayúsculas y se veía mal. Cambiamos todo tipo de cosas para mejor presentación y eficacia.

El programa tiene un buen diseño, no tiene complicación alguna a la hora de ejecutarlo, cumple con todas las características que debía de tener y corre a la perfección, obvio debemos de asignar en el main.c la opción que se desee realizar ya sea 1 o 2.

A mi se me hizo un poco más complicado el hecho de estar manejando los nodos de un lado a otro, creo que fue lo que más tardé en captar.

Hice también algunos diagramas de ayuda para tener una idea de cómo funcionaría la función de WordReverse :



Tuve que hacer este tipo de diagramas para ver cómo funcionaba, y como los tres ciclos se implementaban para que funcionase. Creo que la función que más nos costó implementar fue la de WriteResult, ya que usamos struct, y ninguno lo había utilizado antes, así que costo un poco de trabajo, pero vimos que era de mucha utilidad.

Teníamos un miedo a la hora de poner los do-while, ya que no sabíamos que iba a pasar si al usuario se le ocurría poner el primer nodo en NULL, tuvimos la teoría de que tronaría así que tuvimos que hacer cambios en la mayoría de los ciclos.

Hacíamos códigos externos y después lo implementábamos en nuestro código principal, ya que para poder implementarlo en el código principal debíamos saber que necesitábamos, así que lo hacíamos en otro lado y ya después se hacían los ajustes de acomodo para el código fuente.

El material que fue de bastante utilidad fueron los códigos que realizábamos en clase. Ya que nos basamos en trabajos pasados para poder implementar el nuestro, así que el hecho de agregar comentarios a los trabajos que hacemos en clase es bastante útil.

El último detalle que nos dimos cuenta que debía de tener, era cerrar los archivos.

```
86         Printcadena1a(head); // imprime la cadena final
87     } while ( 1 );
88     fclose( File: file); // cierre del archivo input
89     fclose( File: resultfile); // cierre del archivo result
90 }
```