Funciones Utilizadas en Get_Next_Line:

Concatena linea:

- ✓ **Concatena_linea:** concatena_linea es una función diseñada para combinar dos cadenas de caracteres en una sola cadena (linea1 y linea2) y las concatena. Si linea1 es nula, inicializa. Luego utiliza una función llamada ft_strjoin para concatenar linea1 y linea2. Finalmente, libera la memoria asignada a linea1 y retorna la cadena concatenada.

1. Reserva de memoria para linea1 si es nulo: La función verifica inicialmente sí linea1 es nulo. En caso afirmativo, asigna dinámicamente memoria para un solo carácter y lo inicializa con el carácter NULL terminando '\0', creando así una cadena vacía.
2. Gestión de errores de asignación de memoria: En caso de que la asignación de memoria para linea1 falle, la función devuelve nulo, indicando un error.
3. Concatenación de linea1 y linea2: Utiliza una función llamada ft_strjoin para concatenar las cadenas linea1 y linea2. Esta operación crea una nueva cadena que contiene la combinación de ambas.
4. Liberación de memoria de linea1: Después de la concatenación, la memoria asignada para linea1 se libera utilizando la función free, ya que ya no es necesaria.
5. Retorno del resultado: La función devuelve el puntero a la cadena resultante de la concatenación de linea1 y linea2.

Extraer linea:

- ✓ **Extraer_linea:** Se utiliza para extraer una línea de texto desde el totake_lineas. Recorre el totake_lineas carácter por carácter hasta encontrar un ('\n') o el final de la cadena ('\0'). Luego asigna memoria.

1. Verifica la entrada sí el totake_lineas es NULL o si apunta a una cadena vacía. Esta comprobación es importante para evitar errores al intentar acceder a una cadena no válida.
2. Se busca el carácter de ('\n') o el final de la cadena ('\0') en totake_lineas. Esto es necesario para determinar la longitud de la línea que se va a extraer.
3. Incremento de i si se Encuentra un ('\n') , se incrementa i nuevamente para incluir el carácter de ('\n') en la línea extraída. Es para asegurarse de que el ('\n') se incluya en la línea extraída.
4. Asignación de Memoria para linea: Se asigna memoria dinámica para almacenar

la línea extraída. Suficiente memoria para contener la línea completa, incluido el carácter NULL al final.

5. Copia de la Línea desde totake_lineas a linea: Se copian los caracteres de totake_lineas a linea utilizando un bucle. Esto asegura que la línea se copie correctamente, incluido el ('\n') sí existe.

6. Finalización de linea: Se agrega el carácter NULL al final de linea para garantizar que sea una cadena de caracteres válida. Es esencial para que la línea extraída se pueda manipular correctamente como una cadena de caracteres.

7. Retorno de linea: Se retorna la línea extraída correctamente asignada en memoria dinámica. Permitiendo se utilice en otras partes y garantiza que la memoria asignada dinámicamente se libere correctamente cuando ya no sea necesaria.

Actualizar y buscar ('\n'):

✓ **Actualiza_lineas: S**e utiliza para actualizar el totake_lineas después de haber extraído una línea de texto. Busca la primera ocurrencia de un ('\n') en totake_lineas. Si lo encuentra, asigna memoria para almacenar el resto del texto después del ('\n') , copia este texto en la nueva memoria y libera la memoria asignada a totake_lineas.

1. Se utiliza ft_strchr para encontrar el primer ('\n') en totake_lineas. Esto nos indica el inicio del texto de la próxima línea en totake_lineas.

2. Si no se encuentra, significa que totake_lineas contiene solo una línea y no hay más líneas que extraer. Por lo tanto, liberamos la memoria de totake_lineas y retornamos NULL.

3. Memoria para nextlinea: Se reserva memoria dinámica para nextlinea, que almacenará el resto de la cadena después del ('\n') encontrado. Esto es necesario porque nextlinea contendrá el texto restante después de extraer la línea actual.

4. Copia del Resto de la Cadena en nextlinea: Se copia el texto restante después del ('\n') encontrado en nextlinea. Esto se hace para asegurar que solo se extraiga y almacene el texto restante de totake_lineas para su uso posterior.

5. Finalización de nextlinea: Se agrega un carácter NULL al final de nextlinea. Esto es importante para marcar el final de la cadena, lo que permite tratar nextlinea como una cadena de caracteres válida.

6. Liberación de Memoria: Como hemos extraído una línea completa de totake_lineas y hemos copiado el texto restante en nextlinea, ya no necesitamos el contenido anterior de totake_lineas. Por lo tanto, liberamos la memoria de

totake_lineas.

7. Retorno de nextlinea: Finalmente, retornamos nextlinea, que contiene el texto restante después de la línea extraída. Esto nos permite continuar procesando el texto restante en nextlinea.

GNL Get Next Line:

✓ **Lee linea por linea:** Para obtener la próxima línea del archivo utiliza una variable estática totake_lineas para almacenar el texto que aún no se ha procesado completamente. Lee el archivo en bloques de tamaño BUFFER_SIZE hasta encontrar un ('\n') o llegar al final del archivo. Después de leer cada bloque, concatena su contenido con totake_lineas. Luego busca un s('\n') en totake_lineas. Si lo encuentra, extrae la línea utilizando la función extraer_linea, actualiza totake_lineas utilizando actualiza_lineas y retorna la línea extraída. Si no encuentra un ('\n'), continúa leyendo el archivo. Retorna NULL si no puede leer o si llega al final del archivo.

1. totake_lineas se declara como estática para mantener el estado entre llamadas a la función. Esto permite recordar el estado de lectura entre llamadas sucesivas a get_next_line.

2. nueva se utiliza para almacenar temporalmente los datos leídos del archivo en cada iteración de lectura.

3. linea se utilizará para contener la línea que se extraerá del totake_lineas y se retornará al llamador de la función.

4. Bytes_read se utiliza para almacenar el número de bytes leídos en cada iteración de lectura.

5. Verificación de la Entrada: Se comprueba si FD es un descriptor de archivo válido y si BUFFER_SIZE es un valor válido.

6. Si FD es negativo o BUFFER_SIZE es cero o negativo, no hay datos válidos qué leer, por lo que la función retorna 0.

7. Concatenación de Líneas: Se realiza una lectura del archivo en un bucle mientras se cumpla la condición bytes_read > 0.

8. En cada iteración, se leen datos del archivo en el arreglo nueva. Si se produce un error en la lectura (bytes_read < 0), la función retorna NULL.

9. Los datos leídos se agregan al final del totake_lineas utilizando la función concatena_linea, lo que permite acumular las líneas leídas hasta el momento.

10. Extracción de la Línea: Se extrae una línea completa de totake_lineas utilizando la función extraer_linea.

11. Esta línea se almacena en linea para ser retornada por la función al llamador, después de extraer una línea completa, **se actualiza el totake_lineas** para contener el texto restante que aún no se ha procesado. Esto se logra llamando a la función actualiza_linea, que elimina la línea extraída del totake_lineas y conserva el resto del texto para futuras lecturas.