Reporte

Jose Armando Cerna Villaseñor | Atzin Eduardo cruz briones

REGEX

Proyecto Intermedio

**REGEX Proyecto**

**Introducción**

Como proyecto intermedio para la materia de Algoritmos y Estructura de datos II, se realizó un intérprete de expresiones regulares para facilitar la búsqueda de patrones en textos grandes y complejos. Se aplicaron los conocimientos en String Matching que se adquiridos lo largo del segundo parcial y se implementó el algoritmo Booyer-Moore-Horspool en el lenguaje de Python con algunas variantes, tanto en el cálculo de la tabla de malos emparejamientos como las condiciones que consideran si hubo match o no entre el texto y el patrón.

**Funciones**

* Búsqueda simple: La búsqueda simple consiste en hacer coincidir todos los caracteres del patrón a buscar con una subcadena del texto. Esta búsqueda se realiza con la misma función del asterisco debido a que su comportamiento es similar.
* Asterisco (\*): Comodín válido para cualquier letra.



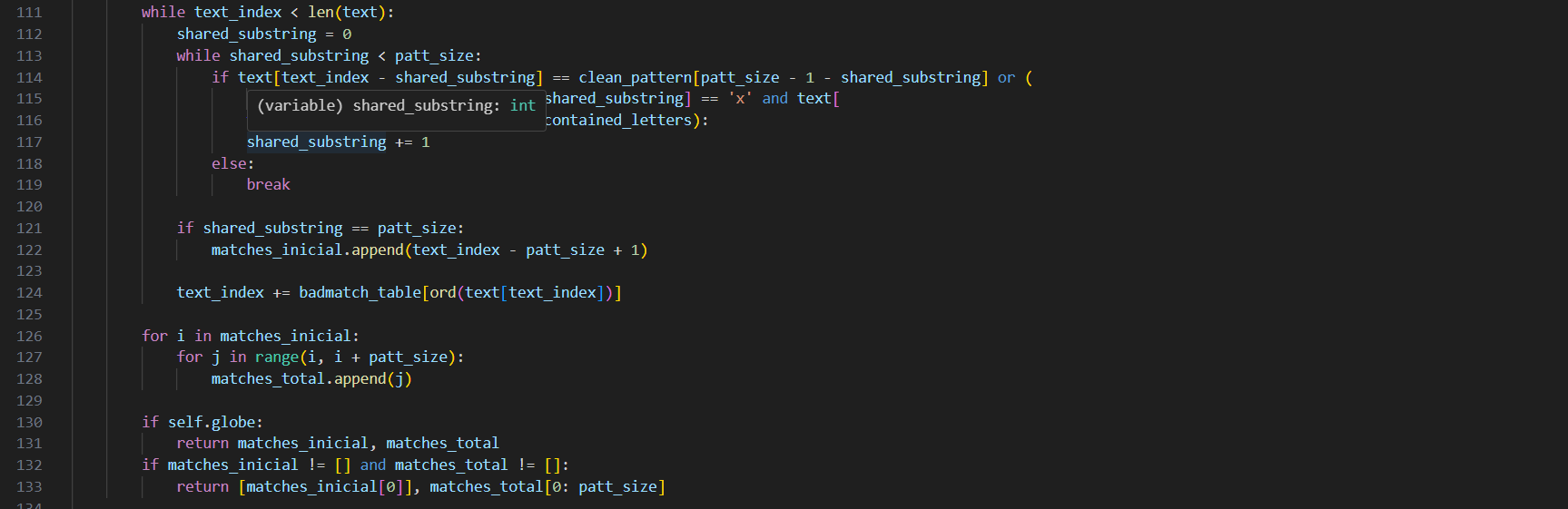
Para el asterisco, al igual que para todas las funciones se implementó el algoritmo de búsqueda Booyer-Moore-Horspool. Se tiene los matches, se calcula la tabla de malos emparejamientos como se hace normalmente y se define nuestro pivote que nos ayudara a comparar el texto con el patrón. Ahora lo que se hace es calcular el índice del patrón en el que se encuentra el asterisco, esto con la finalidad que modificar la tabla de malos emparejamientos y poder hacer los shifts de manera correcta. Creamos un ciclo en el que se itere a lo largo de las letras del alfabeto y si tienen una distancia desde el fin del patrón mayor que la del asterisco se cambia su distancia a la distancia que tiene el asterisco.

Los ciclos de la búsqueda son idénticos a los del algoritmo original, pero a la condición para continuar con la comparación entre un la subcadena del texto y el patrón se modifica un poco; se le agrega que, si el patrón en la posición que se esta comparando es un asterisco, el rango del carácter del texto en ASCII debe ser entre 97 y 122 que corresponde a la a y la z respectivamente, de esta manera sabemos que el asterisco es se compara de manera correcta con una letra y no algún otro carácter como el espacio o el punto.

Al final se devuelven los matches que se fueron acumulando a lo largo de la búsqueda.

* Conjunto de letras:



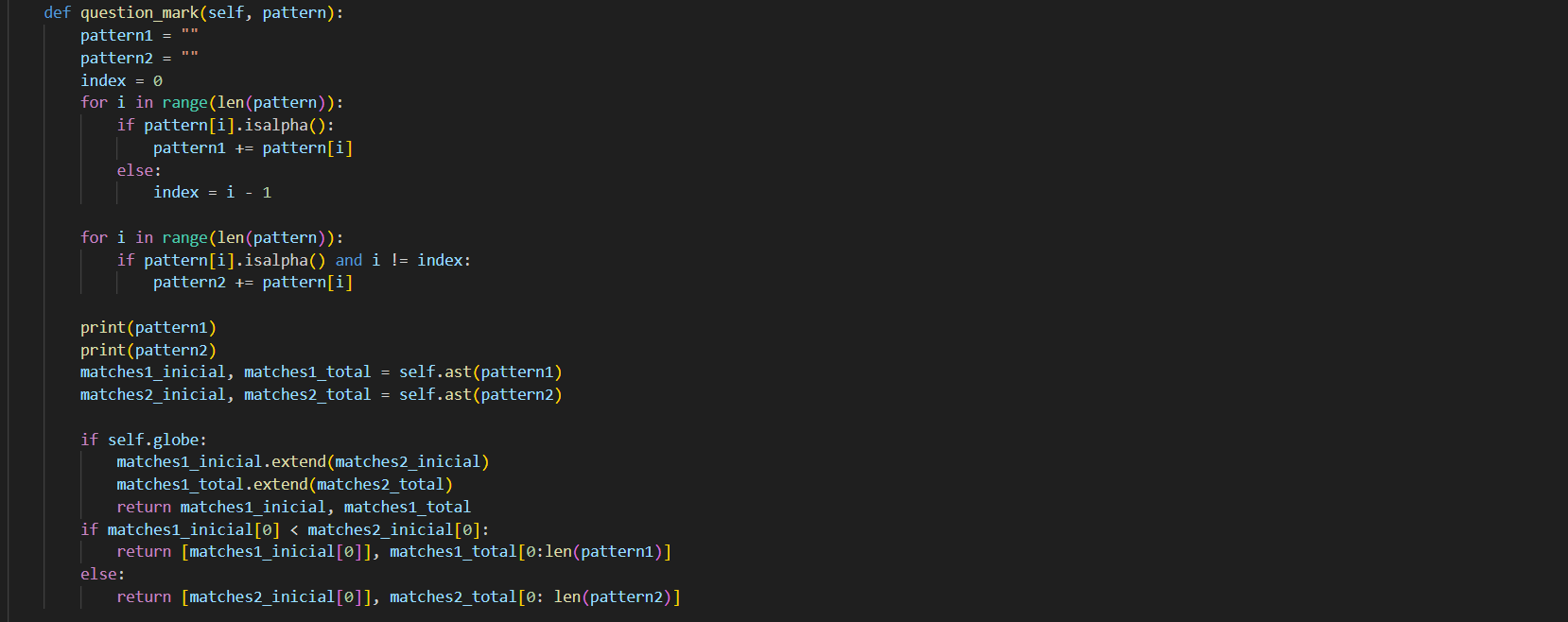


Para las letras que están contenidas entre corchetes, lo que se hizo fue recorrer el arreglo, si el iterador llegaba a la parte del patrón que correspondía al corchete de entrada se activaba una bandera y se guardaba la posición del corchete, al llegar al corchete de salida se desactivaba la bandera. Las letras que se iteraban cuando la bandera estaba activada se guardan en una lista, mientras que las otras letras que se leían cuando la bandera estaba desactivada se guardaban en otra.

Se inserta en la posición donde corresponde el corchete una x.

Se calcula la tabla de malos emparejamientos y se checa si la distancia entre el fin del patrón y la última aparición de la letra dentro del conjunto es mayor a la del corchete de entrada y si es así se cambia por la del corchete. Se verifica se utilizan los mismos ciclos, pero vuelve a cambiar la condición, se hace la comparación igual, pero cuando llegamos al índice al que le corresponde el corchete (donde va la letra contenida en el conjunto), marcada con una x, y se verifica que la letra en el texto este contenida en las letras, de ser así se avanza, y si no se rompe el ciclo de comparación. Al final se devuelven los matches de la búsqueda.

* Signo de pregunta:

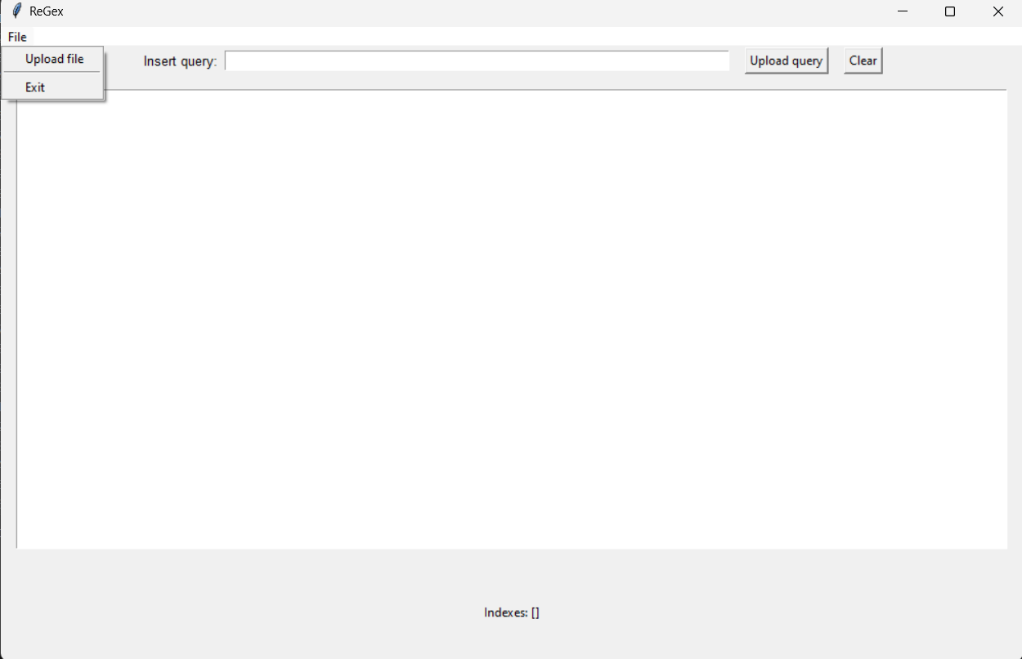


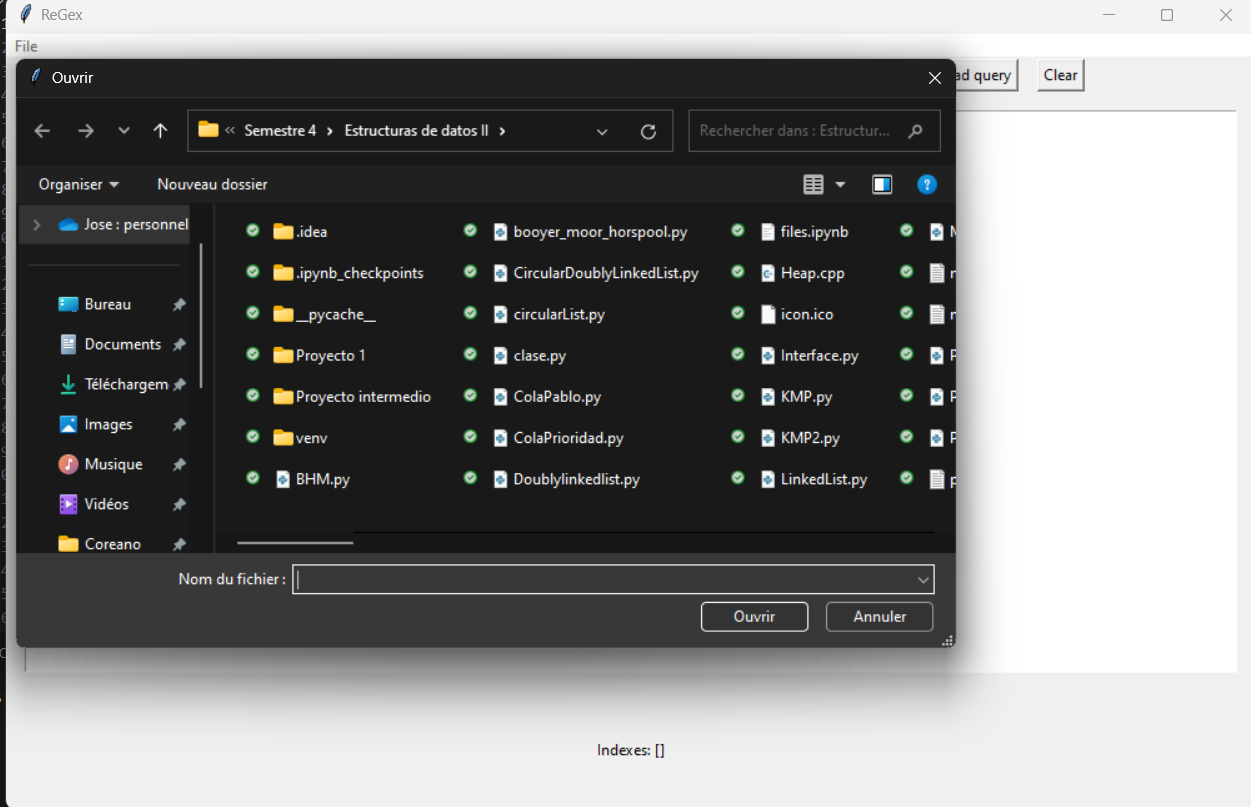
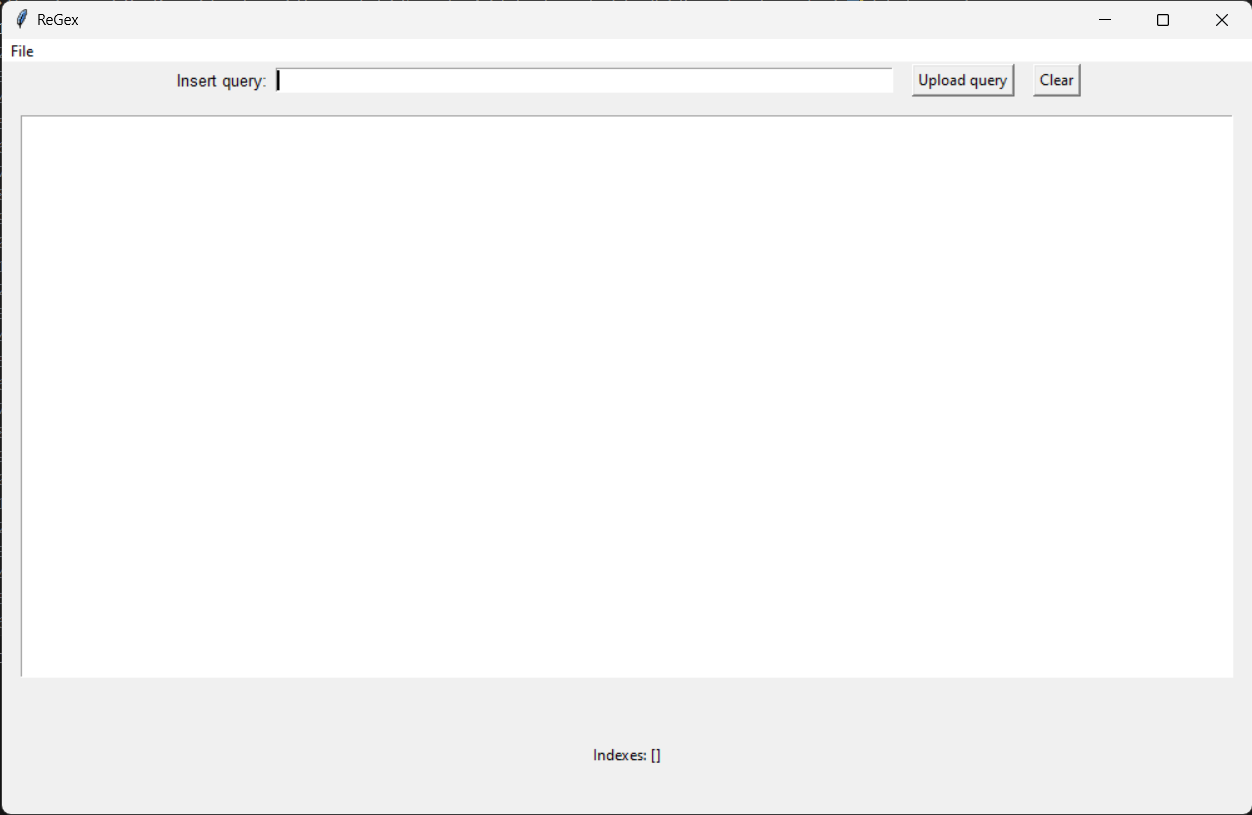
En el signo de pregunta lo que se hace es generar dos patrones, uno donde está el carácter anterior al signo de pregunta y otro donde no. Y se hace una búsqueda simple con ambos patrones. Al final se regresan los matches de ambas búsquedas en un solo arreglo.

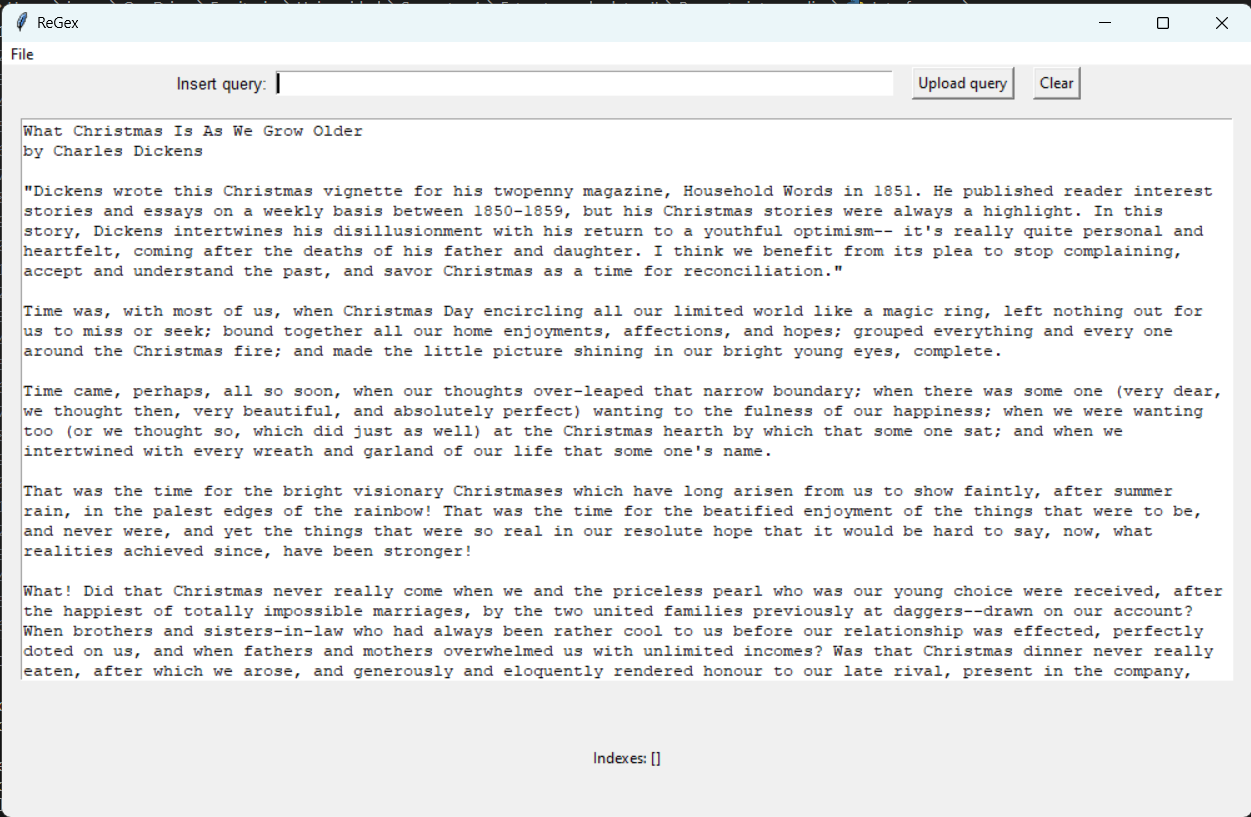
**Interfaz grafica**

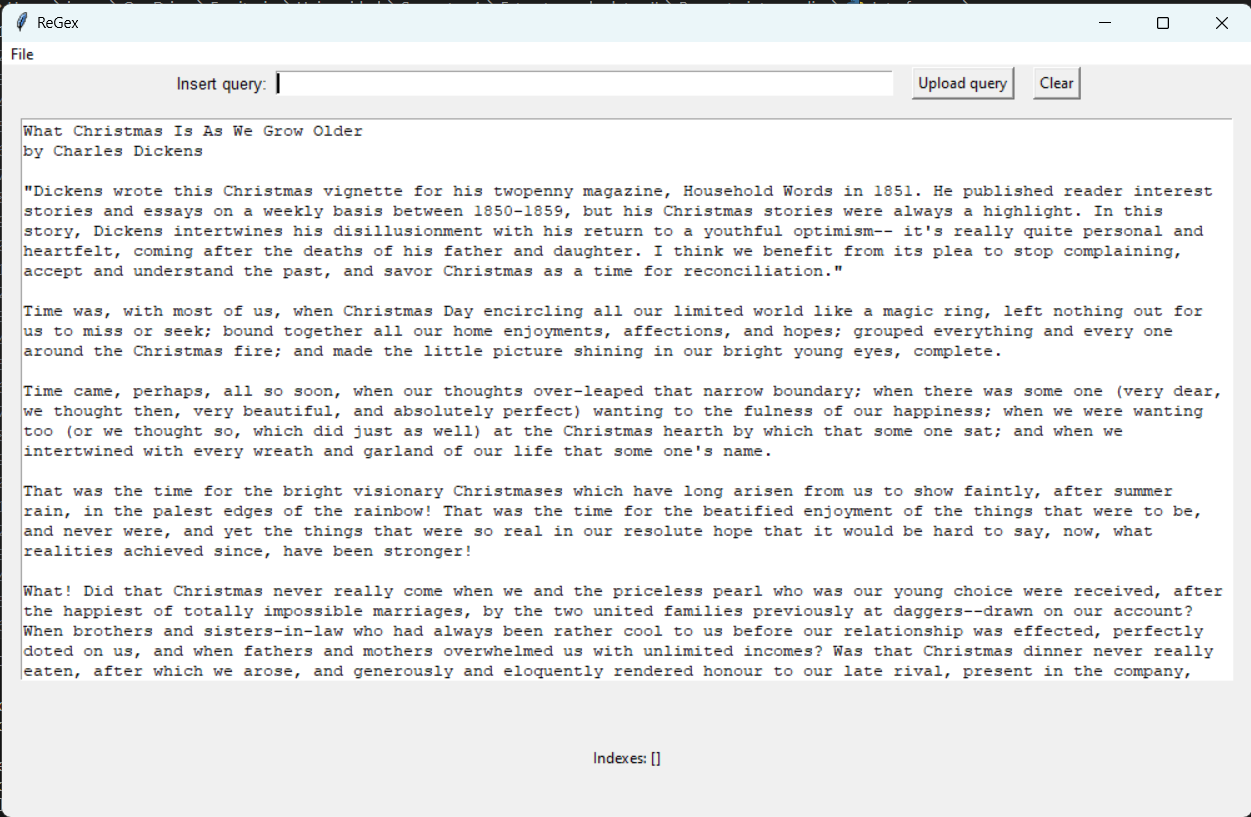
La interfaz grafica cuenta con 4 áreas principales:

* + El área de menú: donde se sube el archivo del que se va a jalar un texto.



* + El área de queries donde podemos mandar la query. Cuenta con el botón de subir query y borrar el input.
  + El área de texto, donde se muestra el texto del archivo que se cargó. Si el archivo a través de la query se llega a modificar, se cargará nuevamente y se mostrará un cuadro de diálogo confirmando su modificación.

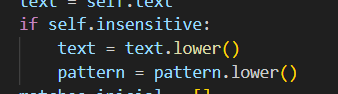


* + Área de índices: Nos muestra el lugar de las incidencias entre el patrón y el texto

**Banderas:**

El proyecto cuenta con dos banderas: ‘g’ que nos índica si se requieren todas las incidencias o ‘i’ si el texto es sensible a mayúsculas y minúsculas.

Para la implementación de la bandera i se pasó tanto el patrón como el texto a minúsculas y las comparaciones se hicieron de la misma manera.



Para la implementación de la bandera g se regresa el arreglo completo de las incidencias y en caso de que no este solo se regresa el primer elemento del arreglo de incidencias. Incluso cuando esta la bandera i activada, el texto se recorre en su totalidad.

**Buscar**

La búsqueda lo que hace es mostrar los arreglos de coincidencias que devuelven cada una de las funciones de los operadores para mostrar su índice en la interfaz.

Esta se hace a través del algoritmo Booyer-Moore-Horspool con distintas modificaciones.

**Buscar y reemplazar**