Practica 03: Diccionario con hashing abierto

**Introducción**

Función Hash

Una función hash, normalmente conocida como “hash” es un algoritmo matemático que transforma cualquier bloque arbitrario de datos en una nueva serie de caracteres con una longitud fija. Independientemente de la longitud de los datos de entrada, el valor hash de salida tendrá siempre la misma longitud.

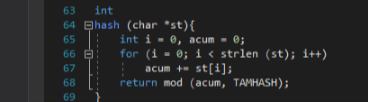
**5 Diccionario con hashing abierto**

**5.1 Planteamiento del problema**

Con la implementación del TAD lista realizar la implementación de una tabla hash abierta, capaz de soportar el almacenamiento de palabras y sus definiciones (Diccionario).

**5.2 Diseño y funcionamiento de la solución**

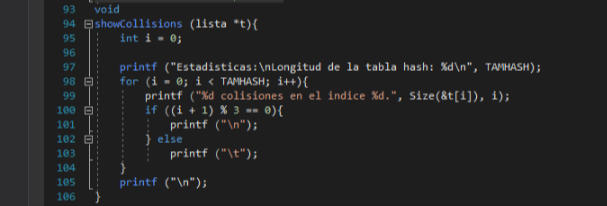
La solución propuesta usa el TAD lista, y fundamentalmente ocupando una función hash en la que, como argumento recibe un puntero a un arreglo de caracteres, solo es relevante el key y, con base a este retorna un digito en donde se corresponde. Está diseñado para de tal forma que los caracteres de key cumplan la condición A <= key <= Z siendo mayúsculas



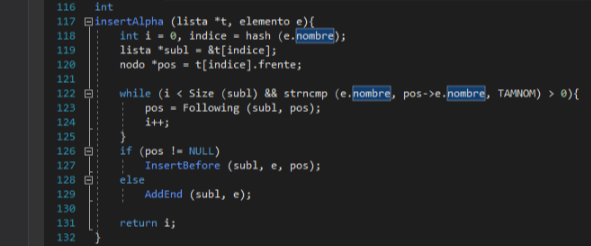
Dado un tablero, sus dimensiones, una columna, el arreglo de soluciones y el número de soluciones; esta función se encargará de buscar todas las configuraciones posibles para colocar a las n reinas en el tablero, usando el método de backtracking, es decir, buscando mediante fuerza bruta todas las soluciones.

**5.3 Implementación de la solución**

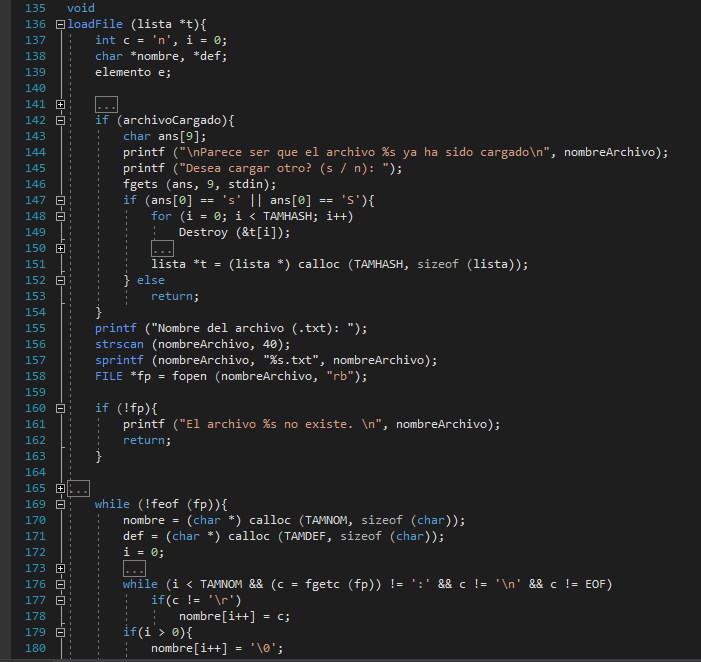
Con funciones como la de mostrar colisiones, muestra las colisiones en la tabla hash. La variable l que sirve para dar un salto de línea cada 3 veces que se muestren las colisiones de una palabra.

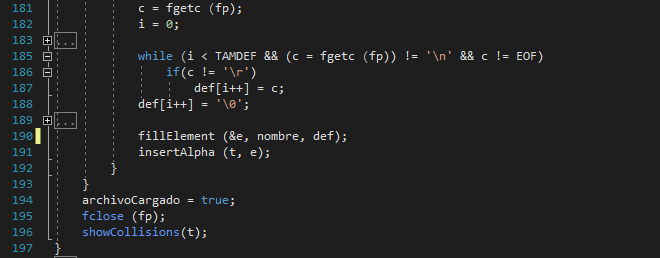


Insertando alfabéticamente en un arreglo de listas un elemento que contenga un campo nombre.

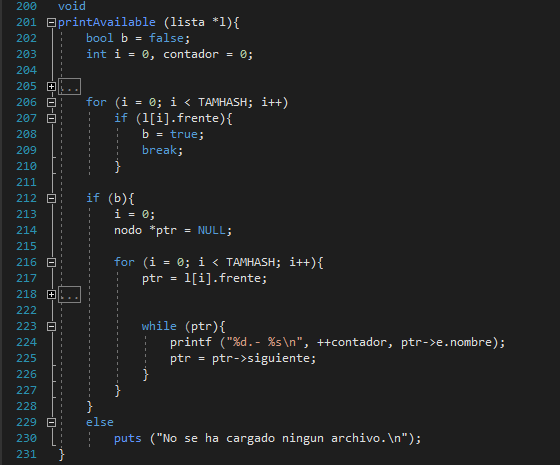


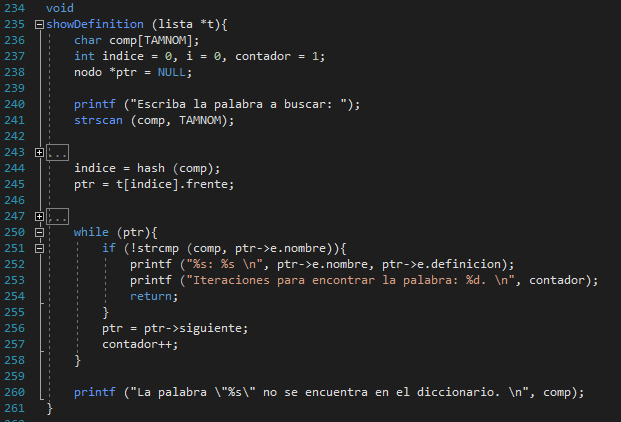
Cargando el archivo y llenando la tabla hash con base a la función.



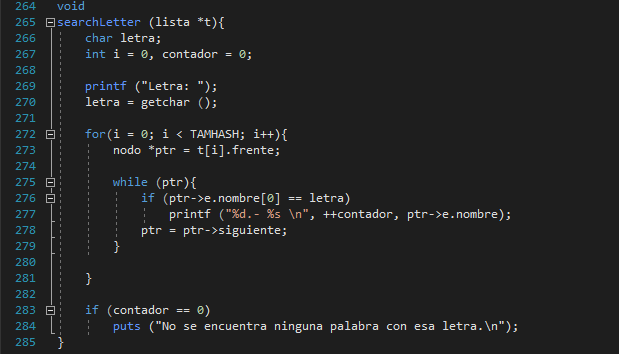


Imprimiendo las palabras del archivo.

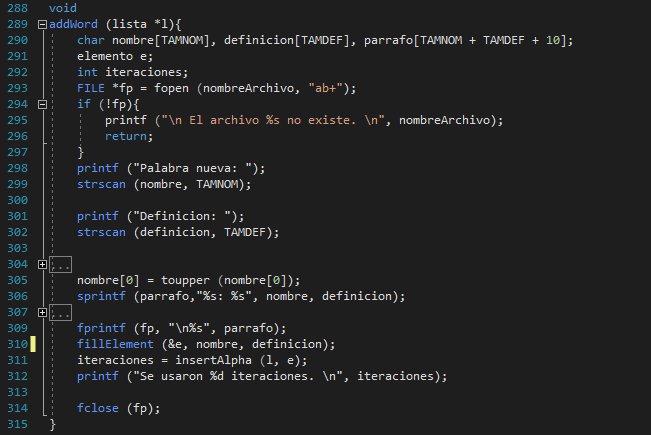


Hace la consulta de una palabra si existe y la imprime.

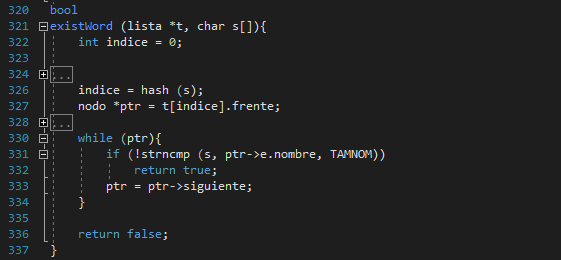
Imprime todas las palabras disponibles con la letra inicial propuesta.



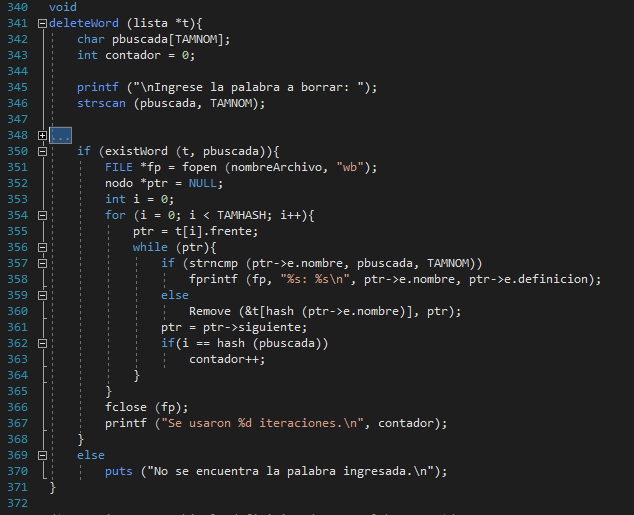
Agrega una palabra, tanto al archivo como a la lista, haciendo que el primer carácter sea mayúsculo si la primera letra es alfabética y escribiendo la palabra al final de nuestro archivo, sin dañar la palabra anterior.

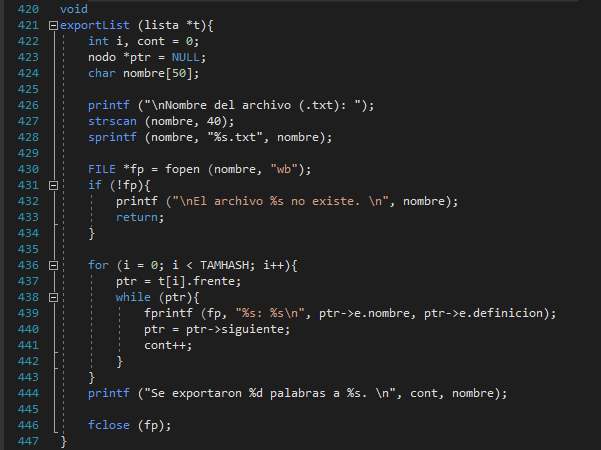


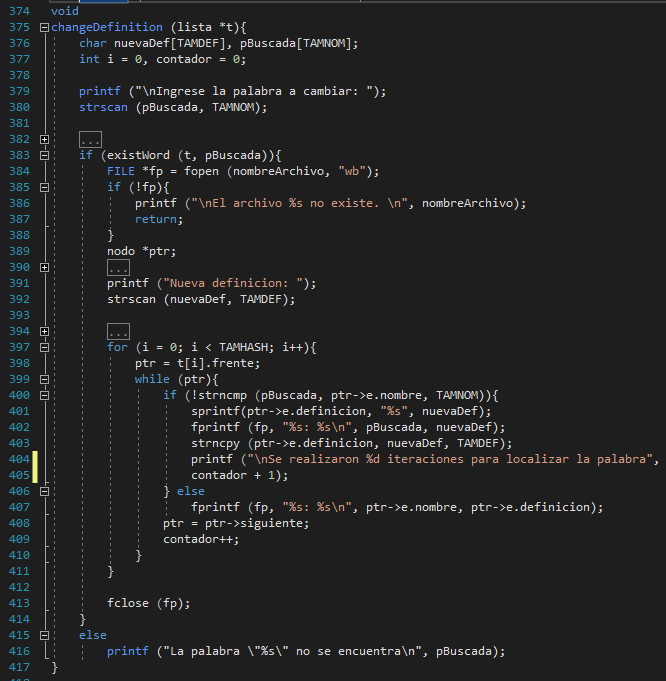
Recibe la lista y una cadena s, en la función solo nos importa la primera letra de la cadena. Como extra cambia el carácter s[0] a mayúscula, en caso de que sea alfanumérico este carácter. Se obtiene el índice de s[0]. y sabiendo que existe un valor correspondiente puesto que ya paso la condición anterior. Buscamos en nuestra fila i-ésima de nuestra tabla, si existe, retorna verdadero, caso contrario falso.



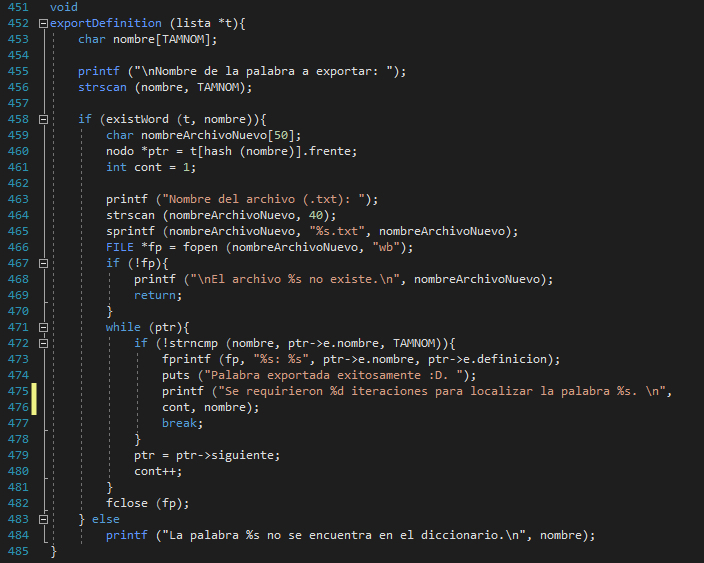
Borra la palabra p, tanto en la tabla como en el archivo, si existe y comprueba que exista la palabra y, si existe, procedemos borramos y escribe el archivo, saltando la palabra no deseada.



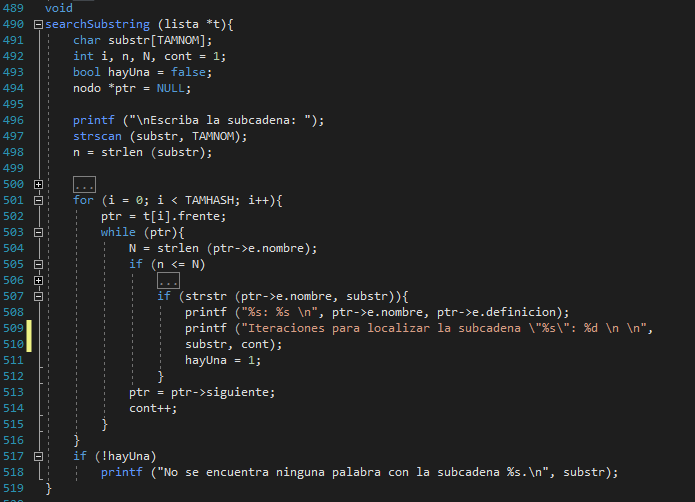
Exporta toda la lista a un archivo con el nombre que el usuario desee.

Cambia la definición de una palabra comprobando si existe la palabra, obteniendo la nueva definición, y recorriendo las 26 palabras reescribe todo el archivo, cuando se encuentra con la palabra a cambiar, escribe la nueva palabra y salta la sobreescritura de la palabra anterior, así como su definición.

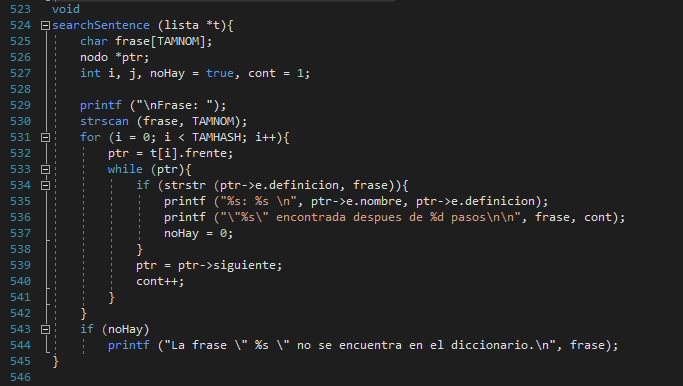
Exporta una palabra con su definición a un archivo de texto guardando la basura al final del archivo creado.



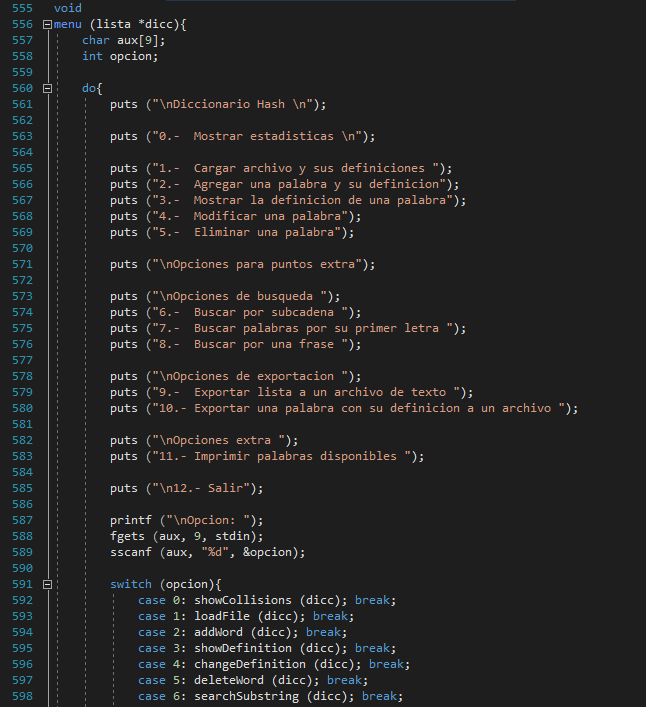
Busca todas las palabras disponibles, con su respectiva definición que contengan una subcadena dada (si existe). Con un bucle recorre las 26 listas que como máximo puede tener la tabla.

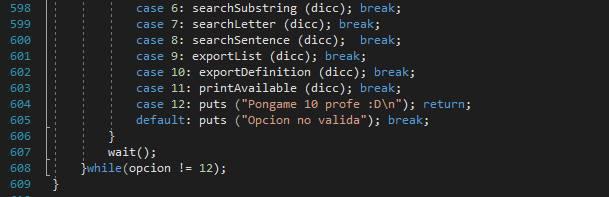


Pregunta alguna frase y muestra las palabras disponibles en el diccionario tales que sus definiciones contengan a la frase.

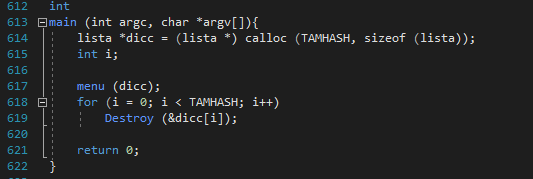


Muestra las opciones disponibles para el programa.





Para al final llamar a las funciones principales.



**5.4 Funcionamiento**

No compila como dice que debe de.

**5.5 Errores detectados**

Cuando se cambia la definición de una palabra se guarda la basura al final del archivo creado.

**5.6 Posibles mejoras**

Como la posible mejora se haría una función que resuelva esto y que se ejecute después de haber creado el nuevo archivo para buscar y si es necesario quitarle lo que no sirva.

**5.7 Conclusiones**

**Pavel**

Mediante una buena función hash eficiente se evitan el mayor número de colisiones en este caso para la función de encontrar las definiciones o palabras de una manera rápida y eficaz y entre más grande sea la búsqueda y o el archivo ahorrar tiempo y recursos.

Fuentes

H. Tiwari, K. Asawa "Cryptographic Hash Function: An Elevated View", European Journal of Scientific Research, 2010

Henk C. A. Van Tilborg,"Encyclopedia of Cryptography and Security" Second Edition. pg 1349. Springer 2011

A. R. Caldebank et all, "Improved Range Summable Random Variable Construction Algorithms", Proceedings of the 16 Annual ACM-SIAM Symposium on Discrete Algorithms