UNIVERSIDAD AUTONOMA DE AGUASCALIENTES

Análisis de Implementación de una Tabla Hash en lenguaje C

Alumno: José Julián Carreón Plascencia

Profesora: Blanca Guadalupe Estrada Rentería

Materia: Compiladores 2

Carrera y curso: ISC 9°A

ID: 244145

1.- Lectura Inicial

El código tiene un tabla de símbolos para un compilador básico, usa una tabla hash con encadenamiento para manejar colisiones. Se definen los datos para poder almacenar las variables con ubicación de memoria y funciones en el mismo código.

2. Identificación de Componentes

Estructura de Datos

Tabla Hash

Es un arreglo de listas enlazadas, con un tamaño fijo de 211. Cada posición de la tabla puede almacenar una lista de variables que colisionan en la misma posición hash.

LineList

Es una lista enlazada simple que guarda los números de línea en los que una variable aparece en el código fuente.

BucketList

Es otra lista enlazada simple que almacena el nombre de una variable, la lista de referencias a líneas donde aparece y su ubicación en memoria.

Funciones Clave

hash(char * key)

Esta función toma un string (nombre de variable) como argumento y devuelve el índice correspondiente en la tabla hash. Genera una posición en la tabla hash para un nombre de variable usando el método de desplazamiento de bits y suma modular. Se desplaza el valor acumulado por un número determinado de bits y luego suma el valor del carácter actual de la cadena, todo esto modulado por el tamaño de la tabla.

st_insert(char * name, int lineno, int loc)

Inserta un nuevo elemento en la tabla de símbolos. Su propósito es almacenar una variable, su ubicación en memoria y las líneas donde aparece. Su implementación es sí, la variable ya está en la tabla, agrega el número de línea; si no está, la inserta en la posición hash correspondiente.

st_lookup(char * name)

Busca una variable en la tabla de símbolos y devuelve su ubicación en memoria. Su principal propósito es encontrar la ubicación de una variable en la tabla de símbolos y se puede calcular midiendo la posición hash de la variable y recorreriendo la lista enlazada de esa posición hasta encontrar la variable o determinar que no existe.

printSymTab()

Imprime la tabla de símbolos completa. También Visualiza todas las variables almacenadas en la tabla, junto con su ubicación en memoria y las líneas en las que aparecen. Se recorre cada posición de la tabla hash e imprime la lista enlazada de cada una.

3. Análisis de la Función Hash

De entrada se recibe un char *, es decir, un string que representa el nombre de la variable. Su salida devuelve un entero (int), que es la posición en la tabla hash donde se almacenará la variable. La implementación está que, en el valor hash, se calcula desplazando los bits de temp a la izquierda y sumando el valor ASCII del carácter actual. Este valor se reduce mediante la operación módulo SIZE para asegurarse de que el resultado esté dentro de los límites de la tabla.

La función utiliza desplazamiento de bits, lo que es computacionalmente eficiente. El uso de módulo por el tamaño de la tabla asegura que el valor esté dentro de los límites, pero la eficiencia en cuanto a la distribución de los valores hash depende de las cadenas de entrada. En general, la función tiene potencial para distribuir bien

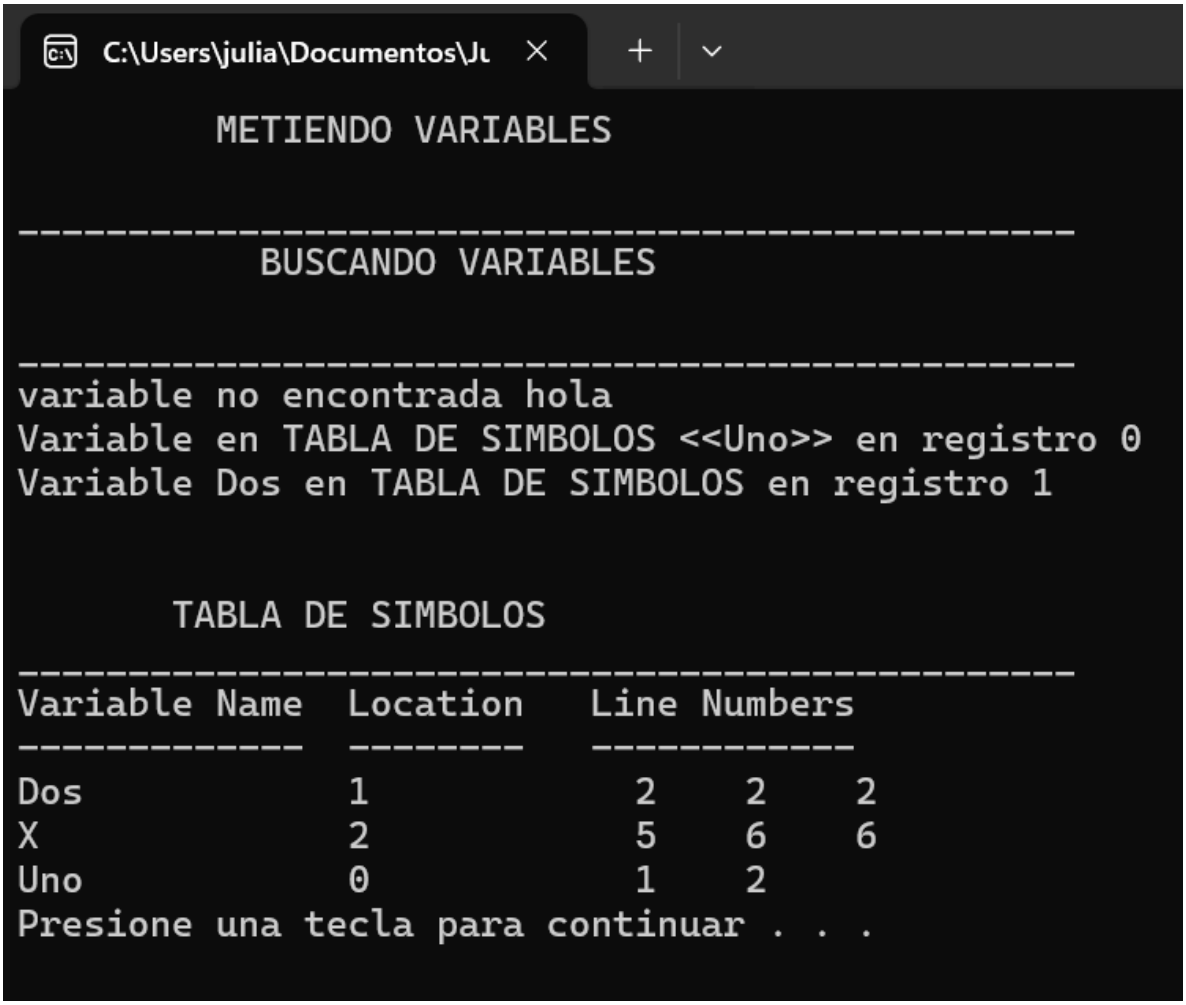
las claves, pero su desempeño en términos de colisiones no puede garantizarse sin pruebas más específicas con diferentes conjuntos de datos.

4. Manejo de Colisiones

El código utiliza **encadenamiento** para gestionar colisiones. Si dos variables generan el mismo valor hash, se almacenan en una lista enlazada en la misma posición de la tabla hash. La función `st_insert` busca si la variable ya existe en la lista y, si no, la agrega al inicio.

Este método es eficiente porque no requiere rehashing ni búsquedas lineales en la tabla, aunque su efectividad depende de mantener las listas enlazadas cortas, lo que está relacionado con la distribución de los valores generados por la función hash.

5. Pruebas



El código en la función main inserta algunas variables y luego las busca. Las variables insertadas son: "Uno", "Dos" y "X", en diferentes ubicaciones y con varios números de línea. Luego se imprimen los resultados de las búsquedas y la tabla de símbolos completa.

La función de inserción (st_insert) parece funcionar correctamente, ya que las variables se insertan y se registran múltiples líneas de referencia. La búsqueda (st_lookup) devuelve las ubicaciones correctas de las variables encontradas y un valor negativo para las variables que no existen, como "hola". El método printSymTab() muestra todas las variables insertadas con sus ubicaciones y líneas, confirmando que el manejo de colisiones también funciona correctamente.

Referencias

- García García, D. (2022). Implementación alternativa de una tabla hash en C++.
- García Hernández, P. (2021). Evaluación empírica de tablas hash.
- Tirado Soto, M. (2021). Desarrollo e implementación de algoritmos para hashing geométrico.