TDA Hash

Se pide implementar un diccionario basado en una **Tabla de Hash cerrada** (direccionamiento abierto) en C. Para ello se brindan las firmas de las funciones públicas a implementar y **se deja a criterio del alumno la creación de las estructuras y funciones privadas del TDA** para el correcto funcionamiento de la **Tabla de Hash** cumpliendo con las buenas prácticas de programación. Para esta implementación las claves admitidas por la tabla serán solamente strings (no nulos). Adicionalmente se pide la creación de un iterador interno que sea capaz de recorrer las claves almacenadas en la tabla.

El TDA entregado deberá compilar y pasar las pruebas dispuestas por la cátedra sin errores, adicionalmente estas pruebas deberán ser ejecutadas **sin pérdida de memoria**.

Para la resolución de este trabajo se debe utilizar una **metodología orientada a pruebas**. A tal fin, se incluye un archivo **pruebas_alumno.c** que **debe** ser completado con las pruebas pertinentes de cada una de las diferentes primitivas del TDA. El archivo de pruebas forma parte de la entrega y por lo tanto de la nota final. Aún mas importante, las pruebas van a resultar fundamentales para lograr no solamente una implementación correcta, si no también una experiencia de desarrollo menos turbulenta.

Parte teórica

Explicr teóricamente los siguientes puntos (no necesariamente en orden, pero por favor usando diagramas):

- Qué es un diccionario - Explicar 3 formas diferentes de implementar un diccionario (tabla de hash cuenta como 1). Justificar ventajas y desventajas.
- Qué es una función de hash y qué características debe tener para nuestro problema en particular
- Qué es una tabla de Hash y los diferentes métodos de resolución de colisiones vistos.
- Explicar por qué es importante el tamaño de la tabla (tanto para tablas abiertas como cerradas)
  - Dado que en una tabla abierta se pueden encadenar colisiones sin importar el tamaño de la tabla, ¿Realmente importa el tamaño?
- Mas te vale que expliques con dibujos