

Centro Universitario el Naranjo, Mixco

Facultad de Ingeniería en Sistemas y Ciencias de la Computación.

Programación III

Ing. José Miguel Villatoro Hidalgo

***Hoja de trabajo 8***

*Luis Eduardo González Alvarado*

*9490-22-14408*

*Keily Andrea Tobar Morales*

*9490-22-4796*

*Sección C*

Guatemala, 25 de mayo de 2024

## Tablas Hash

### ¿Qué son?

Una **tabla hash** es una estructura de datos que implementa un **arreglo asociativo** (o diccionario). Un arreglo asociativo es una colección de pares clave-valor, donde cada clave es única y se utiliza para acceder al valor correspondiente de manera eficiente.

### ¿Cómo funcionan?

Las tablas hash utilizan una función llamada **función hash** para mapear claves a posiciones en una tabla (arreglo) de tamaño fijo. El proceso general es el siguiente:

1. **Inserción**:
   * Se toma la clave del par clave-valor.
   * Se aplica la función hash a la clave para obtener un índice en el arreglo.
   * Se almacena el valor en la posición correspondiente del arreglo.
2. **Búsqueda**:
   * Se toma la clave.
   * Se aplica la función hash para encontrar el índice correspondiente.
   * Se accede a la posición en la tabla para recuperar el valor.
3. **Eliminación**:
   * Similar a la búsqueda, se encuentra el índice usando la función hash y se elimina el valor de la posición correspondiente.

### Conceptos Claves

1. **Función Hash**:
   * Una función que toma una entrada (la clave) y devuelve un número entero, que se utiliza como índice en la tabla.
   * Una buena función hash distribuye las claves de manera uniforme a través de la tabla para minimizar colisiones.
2. **Colisiones**:
   * Ocurren cuando dos claves diferentes se asignan al mismo índice en la tabla.
   * Es inevitable en una tabla de tamaño fijo cuando hay más claves que posiciones posibles.
3. **Factor de Carga**:
   * Es la relación entre el número de elementos almacenados en la tabla y el tamaño de la tabla.
   * Se calcula como α=nm\alpha = \frac{n}{m}α=mn​, donde nnn es el número de elementos y mmm es el tamaño de la tabla.
   * Un factor de carga alto puede llevar a más colisiones y peor rendimiento.

### Técnicas para Evitar Colisiones

1. **Encadenamiento (Chaining)**:
   * Cada posición de la tabla apunta a una lista enlazada que contiene todos los elementos que han sido asignados a esa posición.
   * Las colisiones se manejan añadiendo el nuevo elemento a la lista.
2. **Direccionamiento Abierto (Open Addressing)**:
   * Todos los elementos se almacenan dentro de la propia tabla.
   * Si hay una colisión, se busca otra posición utilizando técnicas como:
     + **Linear Probing**: Se revisan las siguientes posiciones secuenciales hasta encontrar una vacía.
     + **Quadratic Probing**: Se revisan posiciones con incrementos cuadráticos.
     + **Double Hashing**: Se utiliza una segunda función hash para determinar el incremento al buscar una nueva posición.

### Importancia del Factor de Carga en el Rendimiento

* Un factor de carga bajo implica que hay mucho espacio libre en la tabla, lo que generalmente resulta en menos colisiones y una mejor performance.
* Sin embargo, mantener un factor de carga extremadamente bajo puede ser ineficiente en términos de memoria.
* Un factor de carga alto puede causar muchas colisiones, lo que afecta negativamente el rendimiento en términos de tiempo de búsqueda, inserción y eliminación.
* Por lo tanto, es importante encontrar un equilibrio y, a menudo, se utilizan técnicas como **rehashing** (reconstrucción de la tabla con un tamaño mayor) cuando el factor de carga supera un umbral específico.

### Estructuras de Datos Relacionadas

1. **Diccionarios**:
   * En muchos lenguajes de programación, los diccionarios (o mapas) están implementados utilizando tablas hash.
2. **Sets (Conjuntos)**:
   * Los conjuntos, que almacenan elementos únicos, también se implementan comúnmente con tablas hash.
3. **Tries**:
   * Son árboles digitales que se utilizan para almacenar un conjunto de cadenas, donde las cadenas se organizan en nodos del árbol.
   * Aunque no están directamente relacionadas con tablas hash, son otra forma eficiente de almacenar y buscar claves.
4. **Bloom Filters**:
   * Son una estructura de datos probabilística que utiliza múltiples funciones hash para verificar si un elemento pertenece a un conjunto.
   * No permiten eliminación y pueden dar falsos positivos, pero no falsos negativos, lo que las hace eficientes en espacio y tiempo para ciertos tipos de aplicaciones.

### Conclusión

Las tablas hash son una herramienta fundamental en la informática por su eficiencia en las operaciones de búsqueda, inserción y eliminación. Entender cómo funcionan, cómo manejar colisiones y la importancia del factor de carga es crucial para optimizar su rendimiento. Además, conocer estructuras de datos relacionadas amplía las opciones disponibles para resolver problemas específicos de manera eficiente.