Estructuras de Datos

TABLAS DE DISPERSIÓN

ACTIVIDAD A-4.1: CACHÉ DE INFORMACIÓN

René Ornelis

Primer semestre de 2,024

Caché de información

Las tablas de dispersión o tablas de hash son de aplicación general para la búsqueda de elementos, en memoria, de manera casi directa. Una de las aplicaciones más utilizadas de las tablas de dispersión es el uso de caché para bases de datos: una estructura en memoria que contenga los elementos recientes obtenidos de la base de datos, de forma que cuando se busca en elemento, paralelamente se busca en el caché y en la base de datos, de forma que si está en el cache se puede obtener una respuesta más rápida que de la base de datos. De esta forma optimizamos el acceso a la base de datos, ya que el tiempo de acceso de esta es mucho mayor que el acceso a memoria.

1 Objetivos

El objetivo de esta actividad es que el estudiante sea capaz de aplicar los conceptos de tablas de dispersión en una implementación útil.

2 Problema

Le han encargado la implementación de una clase para uso como caché que buscará elementos con una llave numérica y retornará un apuntador a una estructura de información o null si no se encuentra. La definición de la clase a implementar es:

```
typedef struct {
        long CodCliente ;
        string nombre ;
        string telefono ;
        string dirección ;
        long horaAcceso ; //* marca de tiempo al acceder el registro
} Cliente ;

class Cache {
public:
        /** constructor con el número de elementos que se desea obtener */
        Cache (int tam);
        ~Cache() ;
        int size() ; //* número actual de elementos

        /** obtiene la información del cliente o nullptr si no lo encuentra.
        Actualiza la marca de tiempo horaAcceso cuando lo encuentra */
        const Cliente *get (long codCliente) ;
        bool add( Cliente info ) ; //* Retorna true si logra insertar al cliente
        bool del (long codCliente) ; //* Retorna true si logra eliminar al cliente
private:
}
```

Se deben implementar todos los métodos descritos de la clase *Cache*, como una tabla de dispersión, buscando la mejor eficiencia posible, por lo que se deberá realizar la deducción formal de O(n) del método *get*.

2.1 Restricciones

La solución que se entregue debe cumplir con las siguientes restricciones:

- No usar ninguna estructura de datos adicionales
- No utilizar ninguna estructura predefinida en la librería estándar
- No se permite cambiar la visibilidad de los miembros de la clase.
- No debe cambiar la interfaz de las clases (parte pública), aunque sí puede agregar miembros privados.
- Definir solo las variables de clase estrictamente necesarias

3 Tiempo de entrega

La entrega se debe realizar en la plataforma de la Facultad o, en caso de que esta no esté disponible, por correo electrónico, a más tardar el 14/abril a las 23:59. No se permitirá entregas posteriores al límite definido.

4 Entregables

Deberá entregar archivo ZIP con:

1. Archivo .h y .cpp compilables con la definición de clases y la implementación de método solicitado.

5 Criterios de evaluación

Objetivo	Puntos	Detalle por evaluar
Código compilable	0.5	Se entrega código fuente compilable
Variables mínimas	0.5	Agrega solo las variables de clase estrictamente necesarias
Eficacia	2.0	La solución funciona correctamente.
O(n) correcto	0.5	El análisis de O(n) es correcto
Eficiencia	0.5	La solución es eficiente.
TOTAL	4.0	