Práctica 4: Implementación de Tabla Hash

1. Objetivo

Desarrollar en lenguaje C++ una clase genérica que implemente la técnica de búsqueda basada en tabla de hashing con dispersión cerrada.

Realizar un análisis del rendimiento de la tabla de hashing con distintas funciones de dispersión, distintas estrategias de exploración y distintos factores de carga. En este análisis se incluye el estudio del comportamiento para las funciones de dispersión: módulo y pseudo-aleatoria; y para las estrategias de exploración: lineal, cuadrática, dispersión doble y re-dispersión.

2. Entrega

Esta práctica se realizará en dos sesiones de laboratorio en las siguientes fechas:

Sesión tutorada: 5 al 8 de abril de 2016.
Sesión de entrega: 12 al 15 de abril de 2016.

Durante las sesiones de laboratorio se podrán proponer modificaciones y mejoras en el enunciado de la práctica.

3. Enunciado

Desarrollar en lenguaje C++ la plantilla de clases `TablaHash<Clave>` que implemente las siguientes operaciones:

- `Buscar(Clave X)`: retorna el valor booleano `true` si la clave X está guardada en la tabla de hashing. En otro caso retorna `false`.
- `Insertar(Clave X)`: retorna el valor booleano `true` si se añade la clave X a la tabla de hashing. En otro caso retorna `false`.

La clase `TablaHash<Clave>` tiene los siguientes atributos:

- Vector de celdas, `vCeldas`: el número de celdas de la tabla (`nCeldas`) es un parámetro del constructor, que coincide con el número de valores que puede retornar la función de dispersión.
  - Las celdas se implementan mediante la plantilla de clases `Celda<Clave>`, que guarda una cantidad fija (`nBloques`) de valores del tipo `Clave`. El tamaño del bloque es un parámetro de los constructores de la tabla y la celda. La clase `Celda<Clave>` implementa las operaciones:
    - `Buscar(Clave X)`: retorna el valor booleano `true` si la clave X está guardada en el bloque. En otro caso retorna `false`.
    - `Insertar(Clave X)`: retorna el valor booleano `true` si añade la clave X al bloque. En otro caso retorna `false`.
- Función de dispersión, `h(Clave X)`: este miembro implementa la función de dispersión. Recibe como parámetro un valor del tipo `Clave` y retorna una posición dentro del vector de celdas, esto es, un valor entre 0 y `nCeldas-1`. La función de dispersión se especifica mediante un parámetro del constructor de la tabla de hashing.

- Función de exploración, `g(Clave X, int intento)`: este miembro implementa la estrategia de exploración. Recibe como parámetros un valor del tipo `Clave` y el número del intento de exploración. Retorna el desplazamiento (según la estrategia de exploración) respecto a la posición dada por la función de dispersión en la cual se intentará buscar el valor de la clave en dicho intento.

Para realizar el estudio del comportamiento se utilizarán valores de claves del tipo DNI. Un valor de tipo DNI es un número entero de ocho cifras decimales sin letra, con valores entre 30.000.000 y 80.000.000. La clase DNI sobrecarga las operaciones de comparación utilizadas por la plantilla `Tabla<Clave>` cuando `Clave = DNI`.

El programa principal realizará la siguiente secuencia de pasos:

1. Solicita los siguientes parámetros necesarios para instanciar la tabla de hashing:
   a. Número de celdas, `nCeldas`. El número de posiciones de la tabla.
   b. Tamaño del bloque, `nBloques`. El número de claves que se pueden almacenar en cada celda.
   c. Función de dispersión, h(x).
      - Opciones: módulo y pseudo-aleatoria.
   d. Función de exploración, g(x,i).
      - Opciones: lineal, cuadrática, dispersión doble y re-dispersión.
2. Solicita los parámetros del experimento:
   a. Factor de carga, `factor`. Valor entre 0 y 1 que se corresponde al cociente entre el número de valores de clave almacenados y el número de valores que es posible almacenar en la tabla.
   b. Número de pruebas, `nPruebas`. Número de repeticiones de la operación, inserción o búsqueda, que se realiza en el experimento.
3. Crear un banco de prueba con 2*N valores de tipo DNI generados de forma aleatoria. El banco de pruebas se almacena en un vector, con `N=factor*nCeldas*nBloques`.
4. Insertar en la tabla de dispersión los primeros N valores del banco de prueba, hasta alcanzar el factor de carga indicado.
5. El experimento para estudiar el comporamiento de la operación de búsqueda consiste en:
   a. Inicializar a cero los contadores de comparaciones de claves. Valores mínimo, acumulado y máximo.
   b. Realizar la búsqueda de `nPruebas` claves extraidas de forma aleatoria de las primeras N claves del banco de prueba, o sea, de las claves que están guardadas en la tabla de hashing. Para cada búsqueda se cuenta el número de comparaciones de claves realizadas, y se actualizan los valores mínimo, máximo y acumulado.
   c. Al finalizar el experimento se presentan los valores mínimo, máximo y medio del número de comparaciones de claves contabilizados.
6. El experimento para estudiar el comporamiento de la operación de inserción se basa en contar el número de comparaciones para buscar claves que no se encuentran en la tabla. Consiste en:

a. Inicializar a cero los contadores de comparaciones de claves. Valores mínimo, acumulado y máximo.
b. Realizar la búsqueda de `nPruebas` claves extraidas de forma aleatoria de las N claves del banco de prueba que no están guardadas en la tabla hash. Para cada búsqueda se cuenta el número de comparaciones realizadas, y se actualizan los valores mínimo, máximo y acumulado.
c. Al finalizar el experimento se presentan los valores mínimo, máximo y medio del número de comparaciones de claves contabilizados.

A continuación se muestra el formato de salida con los resultados de la ejecución:

Celdas	Bloques	Exploración	Carga	Pruebas
xxxx	xxxx	xxxxxxx	xxxx	xxxx

	Número de Comparaciones		
	Mínimo	Medio	Máximo
Búsqueda	xxxx	xxxx	xxxx
Inserción	xxxx	xxxx	xxxx

Utilizar el programa desarrollado para realizar un estudio de la variación del comportamiento en la tabla de hashing cuando se modifican los parámetros del experimento. Para esto se presentará una gráfica con los valores mínimos, medios y máximos del número de comparaciones al variar el factor de carga entre los valores (0.1, 0.3, 0.5, 0,7, 0.9). De forma similar se puede estudiar la variación del comportamiento al modificar los parámetros número de celdas y número de bloques de la tabla hash manteniendo fijo el factor de carga.

4. Referencias

[1] Apuntes de clase

[2] http://es.wikipedia.org/wiki/Tabla_hash