Universidad de los Andes

Ingeniería de Sistemas y Computación
ISIS1206 – Estructura de Datos
Taller 6

Taller 6: Tabla de Hash

Búsqueda de información de ciudadanos en un directorio de emergencias en caso de colisiones serias de tránsito en una Tabla de Hash.

Objetivos

- ✓ Analizar y comprender la estructura de una tabla de hash.
- ✓ Entender la importancia, necesidad de uso y aplicación de una tabla de hash en aplicaciones de búsqueda, aplicaciones que indexan contenidos, protocolos de seguridad, entre otros.
- ✓ Comprender aspectos básicos de la implementación de una tabla de Hash como el manejo de colisiones.

Descripción General

En Londres se quiere implementar un nuevo proyecto que gestione la información de los ciudadanos afectados en una colisión de tránsito, para lo cual se tienen 2.000.000 registros de los habitantes de esta ciudad. Adicionalmente, el ministerio de transporte y movilidad ingles quiere saber en el menor tiempo posibles los afectados en una colisión de tránsito, estas personas pueden ser buscadas por su nombre, apellido, localización del móvil y número de teléfono del acudiente más cercano.

El alcalde de Londres lo ha contratado para realizar esta aplicación y le ha pedido que cumpla **con los siguientes criterios de acuerdo a la política de libre acceso de la información de contacto:**

- ✓ La aplicación debe tener la información de los habitantes siempre disponible.
- ✓ La aplicación debe buscar la información de la persona asociada a un número de teléfono de forma casi instantánea.
- ✓ Se debe poder ver una lista de todos habitantes con cada uno de sus atributos (nombre, apellido, localización actual, número de contacto del acudiente)
- ✓ La aplicación debe tener la posibilidad de crecer. Es decir que eventualmente se podrían buscar a los ciudadanos por otros criterios.

- ✓ La aplicación debe permitir buscar un habitante con cualquiera de sus atributos (nombre, apellido, localización actual, número de contacto del acudiente)

Abstracción del problema.

El alcalde le ha suministrado un archivo con la información de algunos habitantes de la ciudad de Londres; este archivo contiene la información de 2.000.000 personas (Nombre - Apellido) y el número de teléfono del acudiente más cercano. Asimismo le ha pedido enriquecer los datos con la localización GPS del móvil personal para todos los habitantes.

En específico debe cumplir con los siguientes requerimientos:

R1. Agregar un ciudadano a la lista de emergencias.
R2. Buscar un ciudadano en la lista de emergencias por el número de teléfono de su acudiente.
R3. Buscar un ciudadano por un criterio asociado al Nombre o al Apellido. Decida el criterio de búsqueda en su solución. Bajo este criterio, la respuesta puede ser múltiple.
R4. Buscar un ciudadano por su localización geográfica (latitud y longitud). Recuerde que usted es el encargado de enriquecer con esta información el archivo entregado por el alcalde.

Archivo de entrada.

ciudadLondres.csv - Archivo de datos de los habitantes de la ciudad. Para cada entrada se tienen los datos: nombre*, apellido, teléfono

El nombre es el primer nombre de la persona, No existen registros con nombres compuestos (e.j.: Camilo Andrés,Pérez, 45124544 NO existe)

Note que el archivo de entrada tiene 2,000,000 de triplas.

Lo que usted debe hacer

Parte 1 – Preparación (en casa)

1. Descargue el proyecto **Taller6.zip** impórtelo en Eclipse.
2. Asegúrese de entender bien el problema a resolver descrito en este documento. Modele las clases que necesite para resolver el problema.

3. Analice las clases NodoHash.java y TablaHash.java presentes en la carpeta **estructuras**. ¿Por qué la doble generalidad? Complete las clases anteriormente mencionadas para agregar, buscar y eliminar elementos. (Complete los TODO).
4. Diseñe la estrategia para el manejo de las colisiones cuando la función de hash asigna la misma posición para dos llaves distintas. Decida entre usar el enfoque lineal (*linear probing*) o encadenamiento (*separate chaining*).
5. Implemente la tabla de hash en su modelo del problema. Con esto logrará cumplir con los requerimientos R1y R2.
6. Haga pruebas unitarias tanto de la funcionalidad básica de la tabla de hash como de los requerimientos del mundo.
7. Cree un archivo README.txt en el directorio data. Responda y justifique estas preguntas: ¿Qué pasa si llega nueva información para una llave que ya existe en la tabla de hash? ¿Cuál es el criterio de crecimiento de la tabla? ¿Cuál función de hashing escogió y por qué?
8. Mida el tiempo de ejecución de los requerimientos R1, R2 para tablas de hash con 500000, 1000000, 1500000 y 2000000 de datos. Haga una tabla de comparación de tiempos de R1 y R2 para cada tabla de hash definida ¿Qué opina de estos resultados? ¿Si agrega más datos los tiempos de ejecución cambian significativamente? Responda y justifique estas preguntas en el archivo README.txt

Parte 2 – Trabajo en clase

1. Cree una nueva estructura de datos en la carpeta estructuras para que le permita buscar un objeto por una llave que no es única. En esta estructura para una llave puede tenerse más de un valor asociado. La operación de agregar una pareja (llave, valor) a la nueva estructura de datos permite poder asociar múltiples valores bajo una misma llave. La operación de buscar una llave implica obtener los diferentes valores que estén asociados a la llave. ¿Tiene sentido hacer una tabla de listas?. Responda y justifique en el archivo README.txt
2. Cree pruebas unitarias de la nueva implementación de las estructuras. En específico buscar por llaves no únicas y eliminar por llaves no únicas.
3. Modifique su modelo del mundo para cumplir el requerimiento funcional R3 bajo un criterio de búsqueda que Usted decida: nombre o apellido del ciudadano.
4. Implemente un método que permita obtener las respuestas del requerimiento R3 en orden alfabético. Por ejemplo si busca por nombre, la respuesta debe mostrar en orden alfabético los ciudadanos con el nombre buscado, pero si por el contrario la búsqueda la hace por apellido ésta aparecerá en orden alfabético por apellido.
5. Enriquecer los datos con la localización de los móviles de los ciudadanos.
6. Modifique su modelo del mundo para cumplir el requerimiento funcional R4 bajo un criterio de búsqueda de latitud y longitud.
7. Implemente un método que permita obtener las respuestas del requerimiento R4.
8. Modifique lo necesario para que el requerimiento funcional R1 soporte agregar un ciudadano con localización (latitud y longitud).
9. Modifique lo necesario para que los requerimientos funcionales R2 y R3 entreguen de igual forma los datos de localización (latitud y longitud).
10. Describa en qué situaciones las tablas de Hashing no son una buena opción. Responda y justifique en el archivo README.txt

Entrega

1. Verifique que su proyecto cumple con los requisitos de la entrega de talleres.
2. Responda las preguntas en un archivo README.txt. No olvide agregar este archivo en los archivos a subir a bitbucket.
3. Vuelva a correr las pruebas unitarias asegurándose que éstas evalúan casos críticos y reflejan el correcto funcionamiento de su entrega.
4. Entregue su taller por medio de **BitBucket**. Recuerde, si su repositorio no tiene el taller o está vacío, su taller no podrá será calificado.