

Estructuras de datos y Algoritmos

Lab5-1: Tablas Hash

Objetivo:

- Usar y entender la utilización de la estructura de datos tablas hash,

Material a utilizar

- Para realizar los siguientes ejercicios se necesita de las siguientes clases e interfaces:

Clase *Hashtable*: `java.util.Hashtable`;

Cambio de los nombres de los métodos principales vistos en clase:

`insert` → **`put`**

`find` → **`get`**

`exists` → **`containsKey`**

Para recorrer una *Hashtable* utilizaremos los iteradores tipo `java.util.Enumeration`:

`keys()`: devuelve el iterador con todos los keys de la tabla

`elements()`: devuelve el iterador con todos los datos de la tabla

Interfaz *Entry*: `java.util.Map.Entry`

Clase *LinkedList*, los Interfaces *Iterator* (o *LinkedListIterator*) y *Cola* (`java.util.Queue`)

Ejercicio I

Realizar el ejercicio 1, que se describe a continuación (en la siguiente hoja de este documento), implementando los métodos de la clase *Ejercicio1App.java* del proyecto *Ejercicio1*. En el proyecto también se proveen las siguientes clases:

- *Termino.java*: clase ayudante para definir los elementos de la lista.
- Esta clase tiene que implementar el interfaz `java.util.Map.Entry`, para que su uso en las tablas Hash sea más fácil y el código quede más ordenado.

Nota: Para implementar el interfaz, haz uso del ayudante de Eclipse:

1. *Source* → *Override/Implement Methods*
2. Selecciona el interfaz que quieres implementar (`java.util.Map.Entry`)
3. Implementa los métodos que ha añadido el interfaz: `getKey()`, `getValue()`, `setValue()`

Ejercicio II

Realizar el ejercicio 2, que se describe a continuación (en la 4ª hoja de este documento), implementando los métodos de la clase *Ejercicio2App.java* del proyecto *Ejercicio2*. En el proyecto también se proveen las siguientes clases:

- *ElementoLista.java*: clase ayudante para definir los elementos de la lista
- *ElementoTabla.java*: clase ayudante para definir los elementos de la tabla
- Esta clase tiene que implementar la interfaz `java.util.Map.Entry`, para que su uso en las tablas Hash sea más fácil y el código quede más ordenado.

Ejercicio 1 (3,5 puntos)

Se pide:

- a) *Especificar, diseñar e implementar en Java* un método que dados dos polinomios (representados mediante una lista ligada de términos), calcule y devuelva el resultado de la multiplicación de ambos polinomios. El polinomio resultante se representará mediante una tabla hash.

public static HashTable multiplicaPolinomios(LinkedList pol1, LinkedList pol2);

El tipo de los elementos de la lista será:

```
class Termino {  
    int exp;  
    int coef;  
}
```

Se supone que los términos del polinomio están ordenados de forma decreciente por su grado (valor del exponente) y que las listas dadas no son vacías.

Ejemplo:

para las siguientes listas:

ListaA: $4x^{450} - 8x^{100} + 2$

ListaB: $6x^{100} + 4$

(Resultado: $24x^{550} + 16x^{450} - 48x^{200} - 20x^{100} + 8$)

Se generaría, por ejemplo, para N=9, la siguiente tabla hash como resultado:

Indice	Clave	Datos
1	450	16
2	550	24
3	200	- 48
4	100	- 20
5	0	8
6	vacía	
7	vacía	
8	vacía	
9	vacía	

Nota: la relación entre la clave y el índice depende de la función hash. En este caso es un ejemplo completamente arbitrario, toda la información no tiene porque estar al principio de la tabla.

- b) **Indicar de forma razonada**, suponiendo que todas las operaciones del TAD tabla_hash y Lista Secuencial son de $O(1)$, el **orden** de la operación realizadas en el apartado anterior.

Ejercicio 2 (4 puntos)

Utilizando las clases: LinkedList, HashTable y Queue (ver sus especificaciones al final).

(No hay que implementar ningún método de las clases)

y utilizando una lista de listas secuenciales de palabras, es decir, cada nodo almacena una palabra y la lista de todos sus sinónimos. A continuación se muestra la estructura de cada nodo de la lista:

```
class ElementoLista {  
    String clave;  
    LinkedList sinónimos;  
}
```

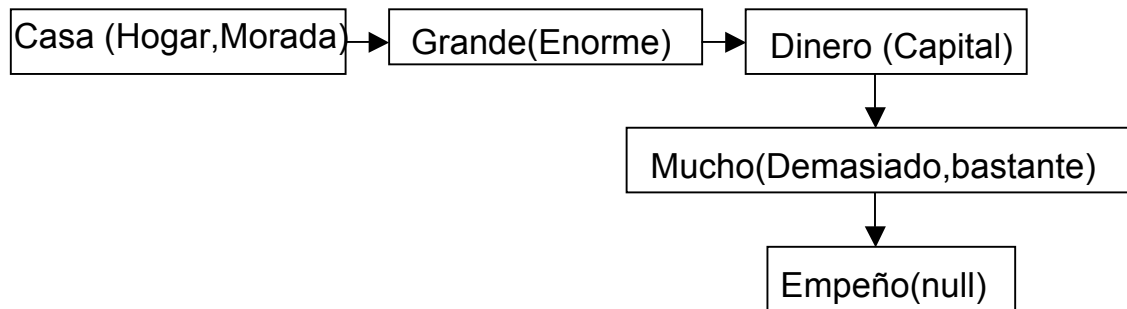
SE PIDE:

(A) Especifica, diseña e implementa en Java un método en la clase lista ligada (LinkedList), que obtenga una tabla hash de colas de palabras sinónimas (ver ejemplo).





```
public HashTable crearTablaHash()
```

Ejemplo:

ENTRADA:



SALIDA:

Indice	Clave	Datos
1	Grande	Enorme 
2	Casa	Hogar Morada 
3	Empeño	(Vacía)
4	(vacía)	
5	Dinero	Capital 
6	(vacía)	
7	Mucho	Demasiado Bastante 
...		
40	(vacía)	

(B) Especifica, Diseña e Implementa en Java un método en la clase tabla hash de sinónimos que dada una lista secuencial de palabras como entrada, escriba en pantalla los sinónimos correspondientes a las palabras dadas.

```
class HashTable {  
    public void visualizarSinonimos(LinkedList palabras)  
}
```

Para ello:

- Si la palabra no está en la tabla hash o si no tiene sinónimos entonces se escribirá como salida la misma palabra
- Si la palabra pertenece a la tabla hash, entonces se escribirá como salida el primer elemento de la cola de sinónimos. Este sinónimo no volverá a ser seleccionado hasta que hayan sido utilizados el resto.

Ejemplo: Para la siguiente lista secuencial de entrada:

(Mi, Casa, Grande, Costo, Mucho, Dinero, Y, Mucho, Empeño)

se escribirá en pantalla :

Mi Hogar Enorme Costo Demasiado Capital Y Bastante Empeño.