TAD HashTable

HashTable = {Par = {Llave <llave>, Valor <valor>}}

{inv: Valor ≠ vacío}

Operaciones Primitivas:

- CrearTabla: -> HashTable
- EstaVacia: HashTable -> Booleano
- LongitudTabla: HashTable -> HashTable
- InsertarTabla: HashTable x Par -> HashTable
- BorrarValorTabla: HashTable x Llave -> HashTable
- DevolverValorTabla: HashTable x Llave -> HashTable
- DarValorHashTabla: HashTable x Llave -> Entero

CrearTabla()

"Crea una tabla hash que esta sin elementos y con un número determinado de ranuras."

{pre: True}

{post: crea una tabla vacía}

EstaVacia(tabla)

"Devuelve valor booleano de verdadero o falso si la tabla está vacía."

{pre: tabla esta creada}

{post: True si la tabla no contiene ningún elemento o Par y
False de lo contrario}

LongitudTabla(tabla)

"Devuelve la cantidad de llaves almacenadas en la tabla."

{pre: tabla esta creada}

{post: entero con valor que indica llaves almacenadas}

InsertarTabla(tabla, par)

"Agrega un Par, que contiene llave y valor, en la tabla."

{pre: tabla esta creada}

{post: inserta el par en la tabla con el valor hash calculado de la llave}

BorrarValorTabla(tabla, llave)

"Borra un par, que contiene llave y valor, en la tabla."

{pre: tabla esta creada}

{post: elimina de la tabla un par el cual coincide con el valor hash de la llave}

DevolverValorTabla(tabla, llave)

“Devuelve valor que almacena la llave en la tabla”

{pre: tabla esta creada}

{post: devuelve el valor de la llave en la tabla}

DarValorHashTabla(tabla, llave)

“Devuelve un entero que identifica la posición o ranura en que se guardara la llave en la tabla”

{pre: tabla esta creada}

{post: devuelve la transformación adecuada a partir de la llave}

Kliver Daniel Girón Castro – Joe Stephen Hernández – Christian Cobo