|  |
| --- |
| TAD Queue |
| Queue: |
| Invariantes: Si la cola está vacía, los punteros head y tail deben ser null. |
| Operaciones Primitivas:   * enQueue: Element 🡪 void * Enqueue 🡪 T * isEmpty 🡪 boolean |
|  |

enQueue()

“add Element in the queue”

{ pre:}

{ post: hasht = {element insert in the queue}

dequeue()

“delete Element in the queue”

{ pre: exist element in the queue }

{ post: hasht = {queue delete the first in the queue }

isEmpty()

“return Boolean the state of the queue”

{ pre: }

{ post: Boolean: true or false}

|  |
| --- |
| TAD HashTable |
| HashTable |
| Invariantes:  Para cada elemento almacenado en la tabla, la posición donde se encuentra debe ser determinada únicamente por su clave y la función hash. |
| Operaciones Primitivas:   * Hash: String * add: element X Key 🡪 void * get: Key X int 🡪 T * printable: 🡪String |
|  |

add()

“Add element in the table”

{ pre: hasht = { table = […] }

{ post: hasht = { table = […, listₖ = [ …, Nodeₘ = key: <key>, value: <value>, …], …]}

Get(key,int)

“Retrieves an element from one of the lists of the table”.

{pre: hasht = {table = […, listₖ = [ …, Nodeₘ = key: <key>, value: <value>, …], …]}

{post:

If key ∈ table: <value>

Else: Null}

printTable()

“print all elements in the table”

{ pre: table != null}

{ post: msj=”Elements in the table”} }

Hash(key)

“Returns a list index to save ”.

{pre: hasht = {table : […]}

{post: <Integer key’s value> % hasht.table.length}

CreateHashtable()

“Creates a new Hashtable object with 0 elements”

{ pre: TRUE}

{ post: hasht = {table: [ ] } }

CreateHashtable()

“Creates a new Hashtable object with 0 elements”

{ pre: TRUE}

{ post: hasht = {table: [ ] } }