TAD <Hashmap>

HashMap = {List<Node<K, T>>}

{inv: There cannot be two elements with the same key on the HashMap.}

Primitive operations:

- HashMap	constructor	HashMap	-> HashMap
- insert	modifier	HashMap x Key x Value	-> HashMap
- delete	modifier	HashMap x Key	-> HashMap
- search	analyzer	HashMap x Key	-> Value
- size	analyzer	HashMap	-> int
- isEmpty	analyzer	HashMap	-> boolean

HashMap(size)
"Creates a new HashMap with the specified initial capacity"
{pre: size $\in$ Integer}
{pos: hash = new HashMap}

insert(key, value)
"Adds an entry with the specified key and value to the map.
If the key already exists, its value is updated."
{pre: hash = {} $\bigwedge$ key $\in$ K $\bigwedge$ value $\in$ V}
{pos: hash = {..., value,...}}

delete(key)
"Removes the entry with the specified key from the map.
If the key does not exist, it does nothing."
{pre: hash = {} $\bigwedge$ key $\in$ K}
{pos: The element of the specified key is removed from
the HashMap.}

search(key)
"Returns the value associated with the specified key.
If the key does not exist, returns null."
{pre: hash = {} $\bigwedge$ key $\in$ K}
{pos: <value> V null}

size()
"Returns the number of elements in the map."
{pre: hash = {}}
{pos: <size>}

isEmpty()
"Returns true if the map is empty, otherwise it returns false."
{pre: hash = {}}
{pos: TRUE V FALSE}