isEmpty()

"Verifies if the stack is empty"

{pre: stack != null}

{post: boolean}

Stack

top == null

boolean = true

getTop()

"Returns the top of the stack"

{pre: stack != null && top != null}

{post: top_value}

Top

-value

TAD STACK

Other Passengers

-LIFO

-attributes

First Passenger

-entranceOrder

-attributes

{inv: LIFO collection type for holding elements: Item Entrance order != Item Exit order && StackElements != null}

+isEmpty() stackTop -> boolean

+getTop() stackTop -> top_value

+setTop() stack x top -> top_node

+push() stack x top -> n/a

+pop() stack x top -> top_node

setTop()

"Assigns a top node to the stack"

{pre: stack != null}

{post: top_node}

Top_node

-value

push()

"Push an element in the stack"

{pre: stack != null}

{post: top_node}

new_Node

-value

pop()

"Pops an element of the stack"

{pre: stack != null}

{post: top_value}

Top_node

-value

isEmpty()

"Verifies if the queue is empty"
{pre: queue != null}

{post: boolean}

Queue

front == null

boolean = true

getFront()

"Returns the front of the queue"
{pre: queue != null && front != null}

{post: front_value}

Front

-value

setFront()

"Assigns a front node to the queue"
{pre: queue != null}

{post: front_node}

Front_node

-value

TAD QUEUE

Other Passengers

-entranceOrder
-attributes

First Passenger

-FIFO
-attributes

{inv: FIFO collection type for holding elements: Item Entrance order == Item Exit order && QueueElements != null}

+isEmpty() queueFront -> boolean
+getFront() queueFront -> front_value
+setFront() queue x front -> front_node
+getLast() queueLast -> last_value
+getSize() queueSize -> size:int
+enQueue() queue x node -> n/a
+deQueue() queue x front -> front_node

getLast()

"Return the last node of the queue"
{pre: queue != null && front != null}

{post: last_value}

Last

-value

getSize()

"Returns the size of the queue"
{pre: queue != null}

{post: int (size)}

Size

-value: int

enQueue()

"Enqueue a node in the queue"
{pre: queue != null}

{post: last_node || front_node}

new_Node

-value

deQueue()

"Dequeue a node of the queue"
{pre: queue != null}

{post: front_value}

Front_node

-value

isEmpty()

"Verifies if the hashtable is empty"
{pre: hashtable != null}

{post: boolean}

HashTable
size == 0
boolean = true

TAD HASHTABLE

←

Hashes
-buckets

Keys
-values

{inv: structure data key type for save and identify elements: keyElement = indexElement && hashElements != null}

+size() hashtable -> size: int
+isEmpty() hashtable -> boolean
+clear() hashtable x hashelements -> n/a
+hash() hashelements x key -> index: int
+put() hashelements

clear()

"Clears the hashtable elements"
{pre: hashtable != null}

{post: boolean}

HashTable
size == 0
boolean = true

size()

"Returns the size of the hashtable"
{pre: hashtable != null}

{post: int (size)}

Size
-value: int

hash()

"Assigns an index to a key"
{pre: hashtable != null}

{post: int (index)}

Key
-key: value
-index: int

put()

"Puts an element in the hashtable"
{pre: hashtable < size}

{post: hash_element}

HashTable
-key: value
-index: int
-hash_element