TESTING DOCUMENTATION

Stack

Scenarios Set Up:

Name	Class	Scenario
setUp1	Stack	New Stack • No elements added
setUp2	Stack	stackActions.push("addTask", "Studying discreetly") stackActions.push("deleteTask", "Studying discreetly") stackActions.push("addTask", "Studying all day") stackActions.push("modifiedTask", "Studying all day", "Going to the gym")

Tests:

Test Objective: Check the correct insertion of the elements in the stack				
Class	**Method**	**Scenario**	**Entry Values**	**Result**
Stack	pushActionInStackEmpty()	setUp1()	"making an integrator"	The stack will just have one element.
Stack	pushActionNotEmptyStack()	setUp2()	"making an integrator"	The element is the last element added so is equal to the top of the stack.

Test Objective: Check if the stack is empty				
Class	**Method**	**Scenario**	**Entry Values**	**Result**
Stack	isEmptyStack()	setUp1()	n/a	True (the stack should be empty)

Stack	notIsEmptyStack()	setUp2()	n/a	False (the stack should not be empty)

Test Objective: Check the correct deletion of the elements in the stack				
Class	**Method**	**Scenario**	**Entry Values**	**Result**
Stack	popActionInStack()	setUp2()	n/a	The tail of the stack should be equal to the popped value.
Stack	popActionInStackEmpty()	setUp1()	n/a	True: An “EmptyStackException” should be thrown.

Test Objective: Check the correct retrieving of the top element in the stack				
Class	**Method**	**Scenario**	**Entry Values**	**Result**
Stack	topActionInStack()	setUp2()	n/a	The tail of the stack should be equal to the top value.
Stack	topActionInStackEmpty()	setUp1()	n/a	True: An “EmptyStackException” should be thrown.

Queue

Scenarios Set Up:

Name	Class	Scenario
setupScenary1	Stack	New queue • No elements added

setUpScenary2	Stack	queue.enqueue("Hola");
		queue.enqueue("Adios");
		queue.enqueue("Buenas tardes");

Tests:

Test Objective: Check the correct insertion of the elements in the queue				
Class	**Method**	**Scenario**	**Entry Values**	**Result**
Queue	enqueueTest()	setUpScenary1 ()	“Hola”	The queue will just have one element: “Hola”

Test Objective: Check the correct deletion of the elements in front of the queue				
Class	**Method**	**Scenario**	**Entry Values**	**Result**
Queue	dequeueTest()	setupScenary2()	“Hola”	The dequeued element should match the expected element: “Hola”.

Test Objective: Check the correct retrieving of the element at the front of the queue without deleting it.				
Class	**Method**	**Scenario**	**Entry Values**	**Result**
Queue	peekTest()	setupScenary2()	n/a	The front element of the queue should match the expected element

Test Objective: Check the isEmpty method works correctly.				
Class	**Method**	**Scenario**	**Entry Values**	**Result**
Queue	isEmptyTest()	setupScenary1()	n/a	True: The queue should be empty
Queue	notIsEmptyTest()	setupScenary2()	n/a	False: The queue should not be empty.

Test Objective: Check if the size method works correctly.				
Class	**Method**	**Scenario**	**Entry Values**	**Result**
Queue	sizeTest()	setupScenary2()	“3”	The size of the queue should be 3

Hash Table

Scenarios Set Up:

Name	Class	Scenario
setupScenary1	HashTable	New queue • No elements added
setUpScenary2	HashTable	hash.add(1, 1)
setUpScenary3	HashTable	hash.add(1, 1); hash.add(2, 2) hash.add(3, 3) hash.add(4, 4) hash.add(5, 5)

Tests:

Test Objective: Check the correct insertion of the elements in the hashTable				
Class	**Method**	**Scenario**	**Entry Values**	**Result**
HashTable	addTest1()	setUpScenary2()	n/a	The hash should not be empty.
HashTable	addTest2()	setUpScenary3()	1, 2, 3, 4, 5	The size of the hash table should match the expected size after adding multiple elements. (size: 5)

Test Objective: Check the isEmpty method works correctly.				
Class	**Method**	**Scenario**	**Entry Values**	**Result**
HashTable	emptyHashTest()	setUpScenary1()	n/a	True: The hash should be empty.

Test Objective: Check the deletion of the elements in hashTable				
Class	**Method**	**Scenario**	**Entry Values**	**Result**
HashTable	deleteEmptyHashTest()	setUpScenary1()	n/a	Deletion from an empty hash table should fail and return false.

Test Objective: Check the search method in a hashTable				
Class	**Method**	**Scenario**	**Entry Values**	**Result**
HashTable	searchTest()	setUpScenary3()	4	The returned value should match the expected value. (4).