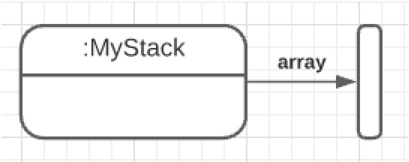
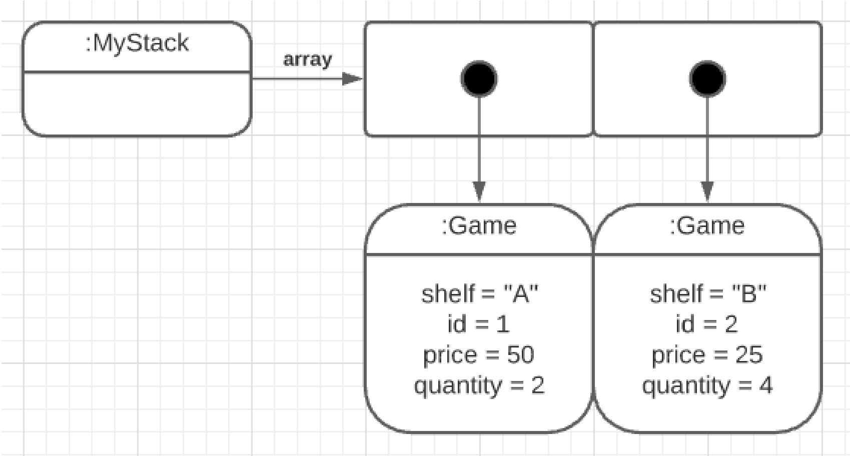
Daniela Bonilla - A00372534, Tomás Ossa - A00372231, Juan Felipe Sinisterra - A00159604

Test Cases Design
Algoritmos y Estructuras de Datos
Tarea Integradora 1

MyStack Unit Cases

Scenes Setup

Name	Class	Scene
sc1	MyStackTest	empty
sc2	MyStackTest	

sc3	MyStackTest	 :MyStack
		array
		:Game
		shelf = "A"
		id = 1
		price = 50
		quantity = 2
		:Game
		shelf = "B"
		id = 2
		price = 25
		quantity = 4

Unit Cases Design

Constructor Method Test

Test Objective: validate the correct creation of a stack of games.				
Class	**Method**	**Scene**	**Entry Values**	**Result**
MyStack	MyStack	sc1	empty	A stack has been successfully created with an empty game arraylist and a top value of -1.

Push Method Tests

Test Objective: validate if an element is added successfully to an empty stack.				
Class	**Method**	**Scene**	**Entry Values**	**Result**
MyStack	push	sc2	Game game = {“Shelf A”, 10, 50, 1}	The game has been successfully added to the top

				of the stack, being the first element on it. The stack top value is now 0 after increasing by one.

Test Objective: validate if an element is added to the top of the stack when the stack has already elements.

Class	Method	Scene	Entry Values	Result
MyStack	push	sc3	Game game = {“Shelf A”, 10, 50, 1}	The game has been successfully added to the top of the stack, being the first element on it. Top value increases by one.

Peek Method Tests

Test Objective: validate if the top element of the stack is returned without removing it.

Class	Method	Scene	Entry Values	Result
MyStack	peek	sc3	empty	The top element of the stack is returned without removing it.

Test Objective: validate that if a stack is empty, the returned value when peeking is null.

Class	Method	Scene	Entry Values	Result
MyStack	peek	sc2	empty	The returned value is null.

Pop Method Tests

Test Objective: validate if the top element of the stack is removed and returned successfully.

Class	Method	Scene	Entry Values	Result
MyStack	pop	sc3	empty	The top element of the stack is returned and removed successfully. Top value decreases by one.

Test Objective: validate if when popping from an empty stack, the returned value is null.

Class	Method	Scene	Entry Values	Result
MyStack	pop	sc2	empty	The returned value is null. Top value stays static.

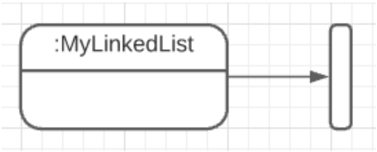
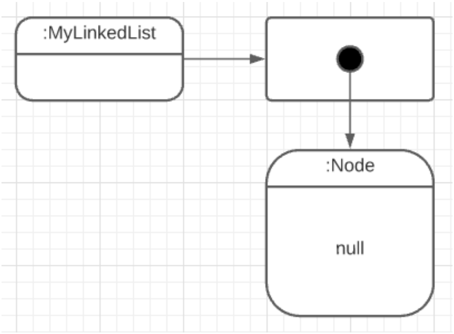
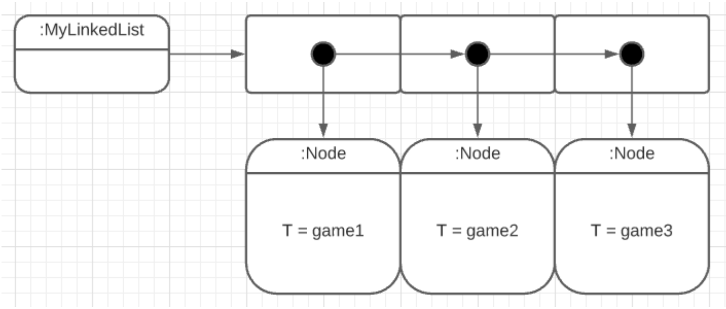
Is Empty Tests

<table>
<tr><td colspan="5"><b>Test Objective:</b> validate if the method returns true when the stack is empty.</td></tr>
<tr><th>Class</th><th>Method</th><th>Scene</th><th>Entry Values</th><th>Result</th></tr>
<tr><td>MyStack</td><td>isEmpty</td><td>sc2</td><td>empty</td><td>Returns true.</td></tr>
</table>

<table>
<tr><td colspan="5"><b>Test Objective:</b> validate if the method returns false when the stack is not empty.</td></tr>
<tr><th>Class</th><th>Method</th><th>Scene</th><th>Entry Values</th><th>Result</th></tr>
<tr><td>MyStack</td><td>isEmpty</td><td>sc3</td><td>empty</td><td>Returns false.</td></tr>
</table>

MyLinkedList Unit Cases

Scenes Setup

Name	Class	Scene
sc1	MyLinkedListTest	
sc2	MyLinkedListTest	
sc3	MyLinkedListTest	 :MyLinkedList :Node T = game1 :Node T = game2 :Node T = game3

Unit Cases Design

Create Node Method Test

Test Objective: validate the correct creation of a node with a game as element.				
Class	**Method**	**Scene**	**Entry Values**	**Result**
MyLinkedList	createNode	sc1	Game game = {“Shelf A”, 10, 50, 1}	Node created successfully.

Insert Node Method Test

Test Objective: validate the correct insertion of a node in the Linked List.				
Class	**Method**	**Scene**	**Entry Values**	**Result**
MyLinkedList	insertNode	sc3	Node newNode = {game4} Node prevNode = node1 Node nextNode = node2	New node inserted successfully in the LinkedList.node

Is Empty Method Test

Test Objective: validate if the method successfully returns true if the Linked List is empty.				
Class	**Method**	**Scene**	**Entry Values**	**Result**
MyLinkedList	isEmpty	sc2	empty	Returns true because the first node is null.
MyLinkedList	isEmpty	sc3	empty	Returns false because the Linked List has elements.

Existing Node Method Test

Test Objective: validate if a given node exists in the Linked List.				
Class	**Method**	**Scene**	**Entry Values**	**Result**
MyLinkedList	existingNode	sc3	Node 1	Returns true because the Node 1 is in the Linked List.
MyLinkedList	existingNode	sc2	Node node4 = {game4}	Returns false because the Node 4 is not in the Linked List.

Delete Node Method Test

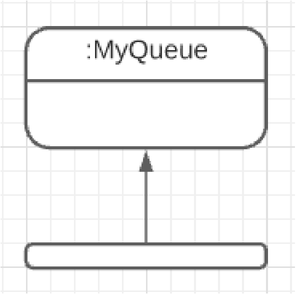
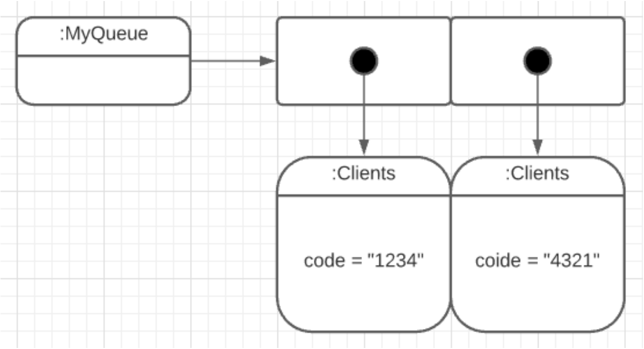
Test Objective: validate if a node is deleted successfully from the Linked List.				
Class	**Method**	**Scene**	**Entry Values**	**Result**
MyLinkedList	deleteNode	sc3	Node 1	Node 1 has been successfully deleted from the Linked List.
MyLinkedList	deleteNode	sc1	Node x = null	Node x was not deleted because it doesn’t exist in the Linked List.

Get Node Method Test

<table>
<tr><td colspan="5"><b>Test Objective:</b> validate if a node is deleted successfully from the Linked List.</td></tr>
<tr><th>Class</th><th>Method</th><th>Scene</th><th>Entry Values</th><th>Result</th></tr>
<tr><td>MyLinkedList</td><td>getNode</td><td>sc3</td><td>Node 2</td><td>Node 2 has been successfully returned.</td></tr>
<tr><td>MyLinkedList</td><td>getNode</td><td>sc2</td><td>Node x = null</td><td>Node x was not retrieved because it doesn’t exist in the Linked List.</td></tr>
</table>

MyQueue Unit Cases

Scenes Setup

Name	Class	Scene
sc1	MyQueueTest	
sc2	MyQueueTest	 :MyQueue :Clients :Clients code = "1234" coide = "4321"

Unit Cases Design

Enqueue Method Test

Test Objective: validate if an element is successfully inserted on the queue.				
Class	**Method**	**Scene**	**Entry Values**	**Result**
MyQueue	enqueue	sc1	Client client = {“1234”}	Client added successfully to the queue. Rear value is now the size decreased by one.

Dequeue Method Test

Test Objective: validate if the front element is successfully returned and deleted from the queue.				
Class	**Method**	**Scene**	**Entry Values**	**Result**
MyQueue	dequeue	sc2		Front element was successfully removed and returned.
MyQueue	dequeue	sc1		Nothing was removed because the queue was empty. Returns null.

Is Empty Method Test

Test Objective: validate if the method returns true if the queue is empty, and returns false if the queue is not empty.				
Class	**Method**	**Scene**	**Entry Values**	**Result**
MyQueue	isEmpty	sc2		Returns false because the queue is not empty.
MyQueue	isEmpty	sc1		Returns true because the queue is empty.

Front Method Test

Test Objective: validate if the method successfully returns the front value of the queue.				
Class	**Method**	**Scene**	**Entry Values**	**Result**
MyQueue	front	sc2		Returns the front element of the queue.
MyQueue	front	sc1		Returns null because the queue is empty.

Rear Method Test

Test Objective: validate if the method successfully returns the rear value of the queue.				
Class	**Method**	**Scene**	**Entry Values**	**Result**
MyQueue	rear	sc2		Returns the front element of the queue.
MyQueue	rear	sc1		Returns null because the queue is empty.

Size Method Test

Test Objective: validate if the method successfully returns the size of the queue.				
Class	**Method**	**Scene**	**Entry Values**	**Result**
MyQueue	size	sc2		Returns the size of the queue that is not empty..
MyQueue	size	sc1		Returns 0 because the queue is empty.