DISEÑO DE PRUEBAS

QUEUE TESTS

Scenarios setup:

Name	Class	Scenario
setup1	QueueTest	Adds a client with the next information: Object: **Account1** account id: 111222 savings: 3.000.000 Object: **CCard1** credit card id: 333444 amount: 100.000 Object: **Client1** Name:Marco Vásquez id: 1006309297 bank account: Account1 credit card: CCard1 entrance date to bank:01/3/2010 disabilities: 0 ---------- Adds a client with the next information: Object: **Account2** account id: 100200 savings: 2.000.000 Object: **CCard2** credit card id: 200300 amount: 300.000 Object: **Client2** Name:David Montoya id: 1005384998 bank account: Account2 credit card: CCard2

		entrance date to bank:01/11/2008 disabilities: 0

Tests cases design:

Test objective: To verify the correct functionality of the method enQueue in the Queue by queueing an object when there is not an object, there is only one and there are two.

Class	Method	Scenario	Input values	Result
Queue	enQueue		Queues a client with the next information: Object: **Account1** account id: 111222 savings: 3.000.000 Object: **CCard1** credit card id: 333444 amount: 100.000 Object: **Client1** Name:Marco Vásquez id: 1006309297 bank account: Account1 credit card: CCard1 entrance date to bank:01/03/2010 disabilities: 0	True if there's only 1 client in the queue
Queue	enQueue		Queues a client with the next information: Object: **Account2** account id: 100200 savings: 2.000.000	True if there are 2 clients in the queue

			Object: **CCard2** credit card id: 200300 amount: 300.000 Object: **Client2** Name:David Montoya id: 1005384998 bank account: Account2 credit card: CCard2 entrance date to bank:01/11/2008 disabilities: 0	
Queue	enQueue		Queues a client with the next information: Object: **Account3** account id: 345123 savings: 200 Object: **CCard3** credit card id: 776677 amount: 500.000 Object: **Client3** Name:Laura Torres id: 1011487566 bank account: Account3 credit card: CCard3 entrance date to bank:22/06/2008 disabilities: 0	True if there are 3 clients in the queue

Test objective: To verify the correct functionality of the method deQueue in the queue, by deQueueing an object when there's more than one object, checking if an object can be added after a removal, removing an object when there's only one and removing in a queue without clients.

Class	Method	Scenario	Input values	Result

Queue	deQueue	setup1	none	True if there's one client in the queue
Queue	deQueue	setup1	EnQueues a client with the next information: Object: **Account3** account id: 345123 savings: 200 Object: **CCard3** credit card id: 776677 amount: 500.000 Object: **Client3** Name:Laura Torres id: 1011487566 bank account: Account3 credit card: CCard3 entrance date to bank:22/06/2008 disabilities: 0	True if there are two clients in the queue
Queue	deQueue	setup1	none	True if there's one client in the queue
Queue	deQueue	setup1	none	True if there's zero clients in the queue
Queue	deQueue	setup1	none	True if the return is null

Test objective: To verify the correct functionality of the method isEmpty in the queue by checking if a queue with clients is empty, if a queue with no clients is empty, if a queue after an enQueue is empty and if that same queue after a deQueue is empty.

Class	Method	Scenario	Input values	Result
Queue	isEmpty	setup1	none	False if it has clients in the queue
Queue	isEmpty	setup1	DeQueues two clients	True if it has no clients in the queue
Queue	isEmpty	setup1	EnQueues a client with the next information: Object: **Account3** account id: 345123 savings: 200 Object: **CCard3** credit card id: 776677 amount: 500.000 Object: **Client3** Name:Laura Torres id: 1011487566 bank account: Account3 credit card: CCard3 entrance date to bank:22/06/2008 disabilities: 0	False if it has clients in the queue
Queue	isEmpty	setup1	DeQueues a client	true if there is no clients in the queue

Test objective: To verify the correct functionality of the method front in the queue, by getting the information of a client when there's various clients, when there's one client and when there's no clients and after a deQueue.

Class	Method	Scenario	Input values	Result
Queue	front	setup1	None	True if the name of the object matches “Marco Vasquez”
Queue	front	setup1	DeQueues a client	True if the name of the object matches “David Montoya”
Queue	front	setup1	DeQueues a client	true if return null
Queue	front	setup1	EnQueues a client with the next information: Object: **Account3** account id: 345123 savings: 200 Object: **CCard3** credit card id: 776677 amount: 500.000 Object: **Client3** Name:Laura Torres id: 1011487566 bank account: Account3 credit card: CCard3 entrance date to bank:22/06/2008 disabilities: 0	True if the name of the object matches “Laura Torres”

STACK TESTS

Scenarios setup:

Name	Class	Scenario
setup1	StackTest	Adds a client with the next information: Object: **Account1** account id: 111222 savings: 3.000.000 Object: **CCard1** credit card id: 333444 amount: 100.000 Object: **Client1** Name:Marco Vásquez id: 1006309297 bank account: Account1 credit card: CCard1 ---------- Adds a client with the next information: Object: **Account2** account id: 100200 savings: 2.000.000 Object: **CCard2** credit card id: 200300 amount: 300.000 Object: **Client2** Name:David Montoya id: 1005384998 bank account: Account2 credit card: CCard2 entrance date to bank:01/11/2008 disabilities: 0

Tests cases design:

Test objective: To verify the correct functionality of the method push in the stack by pushing an object when there is not an object, there is only one and there are two.

Class	Method	Scenario	Input values	Result
Stack	push		Pushes a client with the next information: Object: **Account1** account id: 111222 savings: 3.000.000 Object: **CCard1** credit card id: 333444 amount: 100.000 Object: **Client1** Name:Marco Vásquez id: 1006309297 bank account: Account1 credit card: CCard1 entrance date to bank:01/03/2010 disabilities: 0	True if there's only 1 client in the stack
Stack	push		Pushes a client with the next information: Object: **Account2** account id: 100200 savings: 2.000.000 Object: **CCard2** credit card id: 200300 amount: 300.000 Object: **Client2** Name:David Montoya	True if there are 2 clients in the stack

			id: 1005384998 bank account: Account2 credit card: CCard2	
Stack	push		pushes a client with the next information: Pushes a client with the next information: Object: **Account3** account id: 345123 savings: 200 Object: **CCard3** credit card id: 776677 amount: 500.000 Object: **Client3** Name:Laura Torres id: 1011487566 bank account: Account3 credit card: CCard3 entrance date to bank:22/06/2008 disabilities: 0	True if there are 3 clients in the stack

Test objective: To verify the correct functionality of the method pop in the stack by popping a client in a stack with more than 1 client, checking if it can push after a pop, popping a client in a stack with one client and popping a client in an empty stack.

Class	Method	Scen ario	Input values	Result
Stack	pop	setup 1		True if there's only 1 client in the Stack

Stack	pop	setup 1	Pushes a client with the next information: Object: **Account2** account id: 100200 savings: 2.000.000 Object: **CCard2** credit card id: 200300 amount: 300.000 Object: **Client2** Name:David Montoya id: 1005384998 bank account: Account2 credit card: CCard2	True if there are 2 clients in the Stack
Stack	pop	setup 1	None	True if there's 1 clients in the stack
stack	pop	setup 1	none	True if there are no clients in the stack
stack	pop	setup 1	none	True if the return is null

Test objective: To verify the correct functionality of the method top in the stack by checking if the values are correct in a stack with more than one client, with just one client and no clients, before and after pushing and popping.

Class	Method	Scen ario	Input values	Result
Stack	top	setup 1	none	True if the name of the returned client matches with "David Montoya"

Stack	top	setup 1	Pushes a client with the next information: Object: **Account3** account id: 345123 savings: 200 Object: **CCard3** credit card id: 776677 amount: 500.000 Object: **Client3** Name:Laura Torres id: 1011487566 bank account: Account3 credit card: CCard3 entrance date to bank:22/06/2008 disabilities: 0	True if the name of the returned client matches with “Laura Torres”
Stack	top	setup 1	pop	True if the name of the returned client matches with “David Montoya”
stack	top	setup 1	pop	True if the name of the returned client matches with “Marco Vasquez”
stack	top	setup 1	pop	True if it returns null

HASH TABLE TESTS

Scenarios setup:

Name	Class	Scenario

setup1	HashTableTest	Adds a client with the next information: Object: **Account1** account id: 111222 savings: 3.000.000 Object: **CCard1** credit card id: 333444 amount: 100.000 Object: **Client1** Name:Marco Vásquez id: 1006309297 bank account: Account1 credit card: CCard1 Integer: **Key1** value: Client1 id (1006309297) ---------- Adds a client with the next information: Object: **Account2** account id: 100200 savings: 2.000.000 Object: **CCard2** credit card id: 200300 amount: 300.000 Object: **Client2** Name:David Montoya id: 1005384998 bank account: Account2 credit card: CCard2 entrance date to bank:01/11/2008 disabilities: 0 Intenger: **Key2** value: Client2 id (1005384998)

Tests cases design:

Test objective: To verify the correct functionality of the method insert in the hash table by inserting a client with a key in an empty hash table, in a hash table with one client, and inserting a client that will collide with another one. It's impossible to have clients with the same id in practice, but nevertheless we'll use one for a guaranteed collision.

Class	Method	Scenario	Input values	Result
HashTable	Insert		Inserts a client with the next information: Object: **Account3** account id: 345123 savings: 200 Object: **CCard3** credit card id: 776677 amount: 500.000 Object: **Client3** Name:Laura Torres id: 1011487566 bank account: Account3 credit card: CCard3 entrance date to bank:22/06/2008 disabilities: 0 Integer: **Key3** value: Client3 id (1006782334)	True if the client is found in the array extracted from the hash table
HashTable	Insert		Adds a client with the next information: Object: **Account2** account id: 100200 savings: 2.000.000	True if the client is found in the array extracted from the hash table

			Object: **CCard2** credit card id: 200300 amount: 300.000 Object: **Client2** Name:David Montoya id: 1005384998 bank account: Account2 credit card: CCard2 entrance date to bank:01/11/2008 disabilities: 0 Intenger: **Key2** value: Client2 id (1005384998)	
HashTable	Insert		Adds a client with the next information: Object: **Account1** account id: 111222 savings: 3.000.000 Object: **CCard1** credit card id: 333444 amount: 100.000 Object: **Client1** Name:Marco Vásquez id: 1006309297 bank account: Account1 credit card: CCard1 Intenger: **Key4** value: Client2 id (1005384998)	True if the client is found in the array extracted from the hash table

Test objective: To verify the correct functionality of the search method in the hash table by searching a client with a key in an empty hash table, in a hash table with one client, in a hash table with many clients, and in .

Class	Method	Scenario	Input values	Result
HashTable	search		searches a client with the next key: 1005384998	Throws an exception because there is not a client in the hash table
HashTable	search		Inserts a client with the next information: Object: **Account2** account id: 100200 savings: 2.000.000 Object: **CCard2** credit card id: 200300 amount: 300.000 Object: **Client2** Name:David Montoya id: 1005384998 bank account: Account2 credit card: CCard2 entrance date to bank:01/11/2008 disabilities: 0 Intenger: **Key2** value: Client2 id (1005384998) Searches with the following key: Intenger: **Key2** value: Client2 id (1005384998)	True if the client found matches with the client with key2
HashTable	search		Inserts a client with the next information:	True if the client found matches with the client with key3.

			Object: **Account3** account id: 345123 savings: 200 Object: **CCard3** credit card id: 776677 amount: 500.000 Object: **Client3** Name:Laura Torres id: 1011487566 bank account: Account3 credit card: CCard3 entrance date to bank:22/06/2008 disabilities: 0 Integer: **Key3** value: Client3 id (1006782334) Searches with the following key: Intenger: **Key3** value: Client3 id (1006782334)	
HashTable	search		Searches with the following key: Intenger: **Key4** value: An id not yet registered (1001223451)	False if the client found is null

Tests cases design:

Test objective: To verify the correct functionality of the method delete in the hash table by deleting a client in a hash table with many clients, deleting a client in a hash table with one client, deleting in a hash table without clients, and deleting with a key that's not in the hash table.

Class	Method	Scenario	Input values	Result
HashTable	Delete	setup 1	Deletes the client with the following key: Integer: **Key4** value: An id not yet registered (1001223451)	False if the size of the hash table is different to two
HashTable	Delete	setup 1	Deletes the client with the following key: Integer: **Key1** value: Client1 id (1006309297)	True if the search method doesn't returns Client1
HashTable	Delete	setup 1	Deletes the client with the following key: Integer: **Key2** value: Client2 id (1005384998)	True if the search method doesn't returns Client2
HashTable	Delete	setup 1	Deletes the client with the following key: Integer: **Key2** value: Client2 id (1005384998)	False if an exception is thrown

Test objective: To verify the correct functionality of the method hashFunction in the hash table by checking if the result of the hash function matches with the manual calculation with two different clients. The hash function is h'(k) = k mod m, where k is the key value and m is the size of the hash table.

Class	Method	Scenario	Input values	Result

HashTable	hashFunction		Creates a hash table with 5 slots. Inserts a client with the next information: Object: **Account1** account id: 111222 savings: 3.000.000 Object: **CCard1** credit card id: 333444 amount: 100.000 Object: **Client1** Name:Marco Vásquez id: 1006309297 bank account: Account1 credit card: CCard1 Integer: **Key1** value: Client1 id (1006309297)	True if the result of the function matches with h'(k) = 1006309297 mod 5 = 2
HashTable	HashFunction		Inserts a client with the next information: Object: **Account2** account id: 100200 savings: 2.000.000 Object: **CCard2** credit card id: 200300 amount: 300.000 Object: **Client2** Name:David Montoya id: 1005384998 bank account: Account2 credit card: CCard2 entrance date to bank:01/11/2008 disabilities: 0	True if the result of the function matches with h'(k) = 1005384998 mod 5 = 3

			Intenger: **Key2** value: Client2 id (1005384998)	
HashTable	HashFunction		Inserts a client with the next information: Object: **Account3** account id: 345123 savings: 200 Object: **CCard3** credit card id: 776677 amount: 500.000 Object: **Client3** Name:Laura Torres id: 1011487566 bank account: Account3 credit card: CCard3 entrance date to bank:22/06/2008 disabilities: 0 Integer: **Key3** value: Client3 id (1006782334)	True if the result of the function matches with h'(k) = 1011487566 mod 5 = 1

PRIORITY QUEUE TESTS

Scenarios setup:

Name	Class	Scenario
setup1	PriorityQueueTest	Adds a client with the next information: Object: **Account1** account id: 634188 savings: 80.000.000 Object: **CCard1** credit card id: 000777 amount: 500.000 Object: **Client1** Name: Walter White id: 8005439 bank account: Account1 credit card: CCard1 entrance date to bank:24/07/2004 disabilities: 1 ---------- Adds a client with the next information: Object: **Account2** account id: 412644 savings: 100.000.000 Object: **Client2** Name:Hector Salamanca id: 4651235 bank account: Account2 credit card: Null entrance date to bank:01/11/2001 disabilities: 3 ---------- Adds a client with the next information:

		Object: **Account3** account id: 368512 savings: 40.000.000 Object: **CCard3** credit card id: 546398 amount: 250.000 Object: **Client3** Name: Skyler White id: 7654251 bank account: Account3 credit card: CCard3 entrance date to bank:11/12/2006 disabilities: 2

Tests cases design:

Test objective: To verify the correct functionality of the method insert in the priority queue by inserting a client when the priority queue is empty, when there is only one client in the priority queue and when there are two clients.

Class	Method	Scen ario	Input values	Result
Priority Queue	insert		Adds a client with the next information: Object: **Account1** account id: 634188 savings: 80.000.000 Object: **CCard1** credit card id: 000777 amount: 500.000 Object: **Client1** Name: Walter White id: 8005439 bank account: Account1 credit card: CCard1	True if there's only 1 client in the priority queue

			entrance date to bank:24/07/2004 disabilities: 1	
Priority Queue	insert		Inserts a client with the next information: Object: **Account2** account id: 412644 savings: 100.000.000 Object: **Client2** Name:Hector Salamanca id: 4651235 bank account: Account2 credit card: Null entrance date to bank:01/11/2008 disabilities: 3	True if there are 2 clients in the priority queue
Priority Queue	insert		Adds a client with the next information: Object: **Account3** account id: 368512 savings: 40.000.000 Object: **CCard3** credit card id: 546398 amount: 250.000 Object: **Client3** Name: Skyler White id: 7654251 bank account: Account3 credit card: CCard3 entrance date to bank:11/12/2006 disabilities: 2	True if there are 3 clients in the priority queue

Test objective: To verify the correct functionality of the method extractMax in the queue, by extracting a client when there's more than one client, checking if a client can be added after an extraction, removing a client when there's only one and extracting in a queue without clients.

Class	Method	Scenario	Input values	Result
Priority Queue	extract Max	setup1	none	True if the name of the client returned matches “Hector Salamanca”
Priority Queue	extract Max	setup1	Inserts a client with the next information: Object: **Account2** account id: 412644 savings: 100.000.000 Object: **Client2** Name:Hector Salamanca id: 4651235 bank account: Account2 credit card: Null entrance date to bank:01/11/2008 disabilities: 3	True if there are three clients in the priority queue
Priority Queue	extract Max	setup1	none	True if the name of the client returned matches “Hector Salamanca”
Priority Queue	extract Max	setup1	none	True if the name of the client returned matches “Walter White”
Priority Queue	extract Max	setup1	none	True if the name of the client returned matches “Skyler White”
Priority Queue	extract Max	setup1	none	True if it throws an exception

Test objective: To verify the correct functionality of the method isEmpty in the priority queue by checking if a queue with clients is empty, if a queue with no clients is empty, if a queue after an enQueue is empty and if that same queue after a deQueue is empty.

Class	Method	Scenario	Input values	Result
Priority Queue	Maximum	setup1	none	True if the client returned matches the name “Hector Salamanca” and the size of the priority queue is 3
Priority Queue	Maximum	setup1	ExtractMax	True if the client returned matches the name “Skyler White” and the size of the priority queue is 2
Priority Queue	Maximum	setup1	ExtractMax	True if the client returned matches the name “Walter White” and the size of the priority queue is 1
Priority Queue	Maximum	setup1	ExtractMax	True if it returns null