Computation and discrete structures
Faculty of Engineering

Format of scenarios and use cases

Names::
Leidy Daniela Londoño - A00392917
Isabella Huila Cerón - A00394751

Name	Class	Scenery
setupStage1	PassengersTest	An object of class Passengers with an object of class Passenger - PassengersList - PassengerOne: - name: "Daniel" , chair: "1A" , id: "2" , thousands: 0
setupStage2	PassengersTest	An object of class Passengers with an object of class Passenger - PassengersList - PassengerOne: - name: "Luisa" , chair: "5B" , id: "11" , thousands: 200
setUpStage3	PassengersTest	An object of class Passengers with an object of class Passenger - PassengersList - PassengerOne: - name: "Isabella" , chair: "3A" , id: "4", thousands: 1200

Test Objective: Verify that the Insert method of the HashTable class correctly adds a new passenger, also handling collisions and in case it has repeated, throwing an exception

Class	Method	Scenery	Input Values	Result
HashTable	insert	setupStage1	Passenger: name: "Johan" ,chair: "2B" , id: "6" , thousands: 100	The hash table is now left with two objects, the one that it had that was the passenger Daniel and this new one that entered that the id would be equal to 6.
HashTable	insert	setupStage2	name: "Andrea" ,chair: "2B" , id:: "20" , thousands: 1200	The hash table is left with two objects, the one that was the passenger Daniela and the new one that was entered that would have the id 20. In this case there is a collision, but the function handles it and would allow input

HashTable	insert	setupStage3	name: "Isabella" , chair: "3A" , id: "4", thousands: 1200	The table is left with only one passenger and this new entry is not added, instead it throws an exception indicating that the item is duplicated

Name	Class	Scenery
setupStage4	PassengersTest	An object of class Passengers with three objects of class Passenger - PassengersList - PassengerOne: - name: “Isabella" , chair: "3A" , id: "4", thousands: 1200 - name: "Paula" , chair: "4A", id: "11" , thousands: 0 - name: "Camilo" , chair: "2C" , id: "12" , miles: 10000

Test Objective: Verify that the search method correctly searches for a passenger. Considering if nothing is found and handling collisions				
Class	**Method**	**Scenery**	**Input Values**	**Result**
HashTable	search	setupStage4	name: "Daniela" ,chair: "3C" , id: "20" , thousands: 10000 key: 20	Returns the passenger searched for, in this case, it would be the passenger named Daniela
HashTable	search	setupStage4	key: 20	It does not return anything, since there is no passenger with this key
HashTable	search	setupStage4	name: "John" , chair: "5E" , id: "20" , thousands: 1000 Key: 20	There is a collision, but the find method must handle it correctly. So the method correctly returns the searched value, which would be the passenger with the name Juan

Name	Class	Scenery
setupStage1	PlanePriorityTest	Two objects of class Passengers with one object of class Passenger - PassengersList - PassengerOne: - name: "Isabellal" , chair: "1-A" , id: "1" , thousands: 1000, SpecialNeeds: true, Third age: false - name: "Luisal" , chair: "3-A" , id: "2" , thousands: 100, SpecialNeeds: false, Third age: false.
setupStage2	PlanePriorityTest	An object of class AvionPriorityqueue. null object

setUpStage3	PlanePriorityTest	Two objects of class Passengers with one object of class Passenger - PassengersList - PassengerOne: - name: "Isabellal" , chair: "1-A" , id: "1" , thousands: 1000, SpecialNeeds: true, Third age: false - name: "Luisal" , chair: "3-A" , id: "2" , thousands: 100, SpecialNeeds: false, Third age: false.

Test Objective: Verify that the InsertQueue method of the AvionPriorityQueue class correctly adds a new passenger, taking into account the priority that has been assigned in the hashTable in the data file, and that it is ordered correctly.

Class	**Method**	**Scenery**	**Input Values**	**Result**
PlanePriorityQueue	insertQueue	setupStage1	Passenger: name: "Daniela" ,chair: "2-A" , id: "3" , miles: 1500, SpecialNeeds: false, Third age: true	The priority queue for entering the plane now has three passengers ordered in order of who has the highest priority, in this case it would print Daniela first, then Isabella and finally Luisa.
PlanePriorityQueue	insertQueue	setupStage2	null object	Within the priority queue it is not possible to insert a null passenger
PlanePriorityQueue	insertQueue	setupStage3	Passenger: name: "Daniela" ,chair: "2-A" , id: "3" , miles: 1500, SpecialNeeds: false, Third age: true	The priority queue for departure calculates the priority so that it prints the list for the disembarkation of the plane.

Name	**Class**	**Scenery**

setupStage2	PlanePriorityTest	An object of class AvionPriorityqueue. null object
setUpStage4	PlanePriorityTest	Three objects of class Passengers - PassengersList - name: "Juanl" , chair: "2-C" , id: "1" , thousands: 100, SpecialNeeds: true, Third age: false - name: "Yeison" , chair: "5-A" , id: "2" , thousands: 120, SpecialNeeds: false, Third age: false. - name: "Gabriel" , chair: "3-F" , id: "2" , thousands: 100, SpecialNeeds: true, Third age: true

Test Objective: Verify that the search method successfully removes a passenger. Taking into account, the assigned output.

Class	**Method**	**Scenery**	**Input Values**	**Result**
PlanePriorityQueue	delete	setupStage4	name: "Marinet" ,chair: "3-E" , id: "2" , thousands: 2000, SpecialNeeds: true, Third age: true	The passenger with the highest priority is expected to be eliminated in this case Marinet
PlanePriorityQueue	delete	setupStage4	- name: "Juanl" , chair: "2-C" , id: "1" , thousands: 100, SpecialNeeds: true, Third age: false - name: "Yeison" , chair: "5-A" , id: "2" , thousands: 120, SpecialNeeds: false, Third age: false. - name: "Gabriel" , chair: "3-F" , id: "2" , thousands: 100, SpecialNeeds: true, Third age: true	This test checks if multiple items can be removed from the aircraft priority queue and if the queue is empty after removing those items.
PlanePriorityQueue	delete	setupStage2	initialize the variable	This test tests whether the operation to remove the item with the highest priority in an empty priority queue returns null. First, an empty priority queue is configured using the setup2() method.