**Setup Stages**

|  |  |  |
| --- | --- | --- |
| **Name** | **Class** | **Stage** |
| setup1 | HashTest | A Hash object without values. |
| setup2 | HashTest | An object of the Hash Class with two objects of the Passenger Class:   * First with: name = "Miguel", lastName = "Torres", identification = "12345", PassengerClass = ECONOMY, seat = "A8", age = 18, accumulatedMiles = 50, specialAttention = false, checked = false * Second with: name = "Daniel", lastName = "Gonzales", identification = "23456", PassengerClass = FIRST\_CLASS, seat = "B1", age = 22, accumulatedMiles = 1200, specialAttention = false, checked = false |
| setup3 | HashTest | An object of the Hash Class with five objects of the Passenger Class:   * First with: name = "Miguel", lastName = "Torres", identification = "12345", PassengerClass = ECONOMY, seat = "A8", age = 18, accumulatedMiles = 50, specialAttention = false, checked = false * Second with: name = "Daniel", lastName = "Gonzales", identification = "23456", PassengerClass = FIRST\_CLASS, seat = "B1", age = 22, accumulatedMiles = 1200, specialAttention = false, checked = false * Third with: name = "Maria", lastName = "Gomez", identification = "34567", PassengerClass = FIRST\_CLASS, seat = "C2", age = 40, accumulatedMiles = 150, specialAttention = true, checked = false * Fourth with: name = "Juan", lastName = "Perez", identification = "45678", PassengerClass = ECONOMY, seat = "B5", age = 33, accumulatedMiles = 0, specialAttention = false, checked = false * Fifth with: name = "Javier", lastName = "Roman", identification = "56789", PassengerClass = ECONOMY, seat = "C6", age = 75, accumulatedMiles = 3000, specialAttention = true, checked = false |
| setup1 | PriorityQueueTest | An object of the PriorityQueue Class without values. |
| setup2 | PriorityQueueTest | An object of the PriorityQueue Class with three objects of the Passenger Class:   * First with: name = "Santiago", lastName = "santiesleo", identification = "23456", PassengerClass = FIRST\_CLASS, seat = "B1", age = 22, accumulatedMiles = 1200, specialAttention = false, checked = false, and key = 40. * Second with: name "Esteban", lastName = "EstebanGZam", identification = "34567", PassengerClass = FIRST\_CLASS, seat = "C2", age = 40, accumulatedMiles = 150, specialAttention = true, checked = false, and key = 30. * Third with: name = "Juan", lastName = "jdColonia", identification = "12345", PassengerClass = ECONOMY, seat = "A8", age = 18, accumulatedMiles = 50, specialAttention = false, checked = false, and key = 50. |
| setup3 | PriorityQueueTest | An object of the Class PriorityQueue with five objects of the Passenger Class:   * First with: name = "Juan", lastName = "jdColonia", identification = "12345", PassengerClass = ECONOMY, seat = "A8", age = 18, accumulatedMiles = 50, specialAttention = false, checked = false and key = 42 * Second with: name = "Santiago", lastName = "santiesleo", identification = "23456", PassengerClass = FIRST\_CLASS, seat = "B1", age = 22, accumulatedMiles = 1200, specialAttention = false, checked = false and key = 23 * Third with: name = "Esteban", lastName = "EstebanGZam", identification = "34567", PassengerClass = FIRST\_CLASS, seat = "C2", age = 40, accumulatedMiles = 150, specialAttention = true, checked = false and key = 56 * Fourth with: name = "Javier", lastName = "Roman", identification = "56789", PassengerClass = ECONOMY, seat = "C6", age = 75, accumulatedMiles = 3000, specialAttention = true, checked = false and key = 56 * Fifth with: name = "Juan", lastName = "Perez", identification = "45678", PassengerClass = ECONOMY, seat = "B5", age = 33, accumulatedMiles = 0, specialAttention = false, checked = false and key = 42 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Verify that the insert method of the Hash class works correctly by successfully inserting objects of the Passenger class. | | | | |
| **Class** | **Method** | **Stage** | **Input values** | **Expected result** |
| Hash | insert | setup1 | identificationPassenger1 = “12345”, passenger1  identificationPassenger2 = “23456”, passenger2 | The two passengers have been successfully added to the Hash table and the size of the Hash table is equal to 2. |
| Hash | insert | setup3 | passenger6 = (name = "Esteban", lastName = "GZam", identification = "54321", PassengerClass = FIRST\_CLASS, seat = "A1", age = 18, accumulatedMiles = 2000, specialAttention = false, checked = false)  identificationPassenger6 = “54321” | The passenger is not added to the Hash table because all its containers are already full. Therefore, it throws an Exception, which informs that the maximum capacity of the Hash table has been exceeded. |
| Hash | insert | setup2 | passenger6 = (name = "Juan", lastName = "jdColonia", identification = "12345", PassengerClass = FIRST\_CLASS, seat = "A3", age = 18, accumulatedMiles = 3500, specialAttention = false, checked = false)  identificationPassenger6 = “12345” | As there is already a passenger with that same identification, an exception should be thrown indicating that it is a duplicate key. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Verify that the search method of the Hash class works correctly, successfully finding an object of the Passenger class. | | | | |
| **Class** | **Method** | **Stage** | **Input values** | **Expected result** |
| Hash | search | setup2 | passenger6 = (name = "Santiago", lastName = "santiesleo", identification = "98765", PassengerClass = FIRST\_CLASS, seat = "B2", age = 19, accumulatedMiles = 1500, specialAttention = false, checked = false)  identificationPassenger6 = "98765" | After entering the key 98765, the search method successfully returns the passenger object (passenger6) that corresponds to that key. |
| Hash | search | setup1 | identificationPassenger1 = “12345” | Since the Hash table is empty, the search method will not find any passenger object that corresponds to the entered key. Therefore, the search method should return null. |
| Hash | search | setup3 | identificationPassenger = “12345” | As there is no object with the given identification as key, the search method will not find it and should return null. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Verify that the delete method of the Hash Class works correctly, successfully deleting an object of the Passenger Class. | | | | |
| **Class** | **Method** | **Stage** | **Input values** | **Expected result** |
| Hash | delete | setup1 | identificationPassenger1 = "12345", passenger1 | After removing the only element in the Hash table, its size should be equal to 0. |
| Hash | delete | setup3 | identificationPassenger = “46789”  identificationPassenger = "98765" | As no objects with the given keys are present in the Hash table, the delete method will not find any objects to delete and hence it will not delete any elements, and the size of the Hash table should remain 5. |
| Hash | delete | setup3 | identificationPassenger = “12345”  identificationPassenger = “23456”  identificationPassenger = “45678” | After deleting 3 passengers that are stored in the Hash table, when executing the search method for each of the deleted passengers, it should return null for each search and the size of the Hash table should be 2. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Verify that the insert method of the Priority Queue Class works correctly, successfully inserting an object of the Passenger Class. | | | | |
| **Class** | **Method** | **Stage** | **Input values** | **Expected result** |
| PriorityQueue | insert | setup1 | 42, passenger1 | After inserting an element into an empty priority queue, its size should be equal to 1. Moreover, when extracting the element with the highest priority, it should return the only passenger that was inserted, as there are no more elements in this priority queue. |
| PriorityQueue | insert | setup1 | 42, passenger1  23, passenger2  56, passenger3 | After inserting 3 elements into the Priority Queue, its size should be equal to 3 and upon extracting the largest element, it should be passenger3. |
| PriorityQueue | insert | setup1 | 42, passenger1  56, passenger2  56, passenger3  23, passenger4 | The size of the Priority Queue should be equal to 4, and upon extracting the element with the highest priority, it should return passenger2 since it was the first to be inserted. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Verify that the maximum method of the Priority Queue class works correctly by successfully returning the Passenger object with the highest priority in the queue without removing it. | | | | |
| **Class** | **Method** | **Stage** | **Input values** | **Expected result** |
| PriorityQueue | maximum | setup2 | passenger1 | It should return that the element with the highest priority among the elements is Passenger1. |
| PriorityQueue | maximum | setup1 | None | It should return null since there are no elements in the Priority Queue |
| PriorityQueue | maximum | setup2 | passenger1  passenger2  passenger3 | Initially, the element with the highest priority is Passenger1. After Passenger1 is extracted, the element with the highest priority should be Passenger2. Finally, after Passenger2 is extracted, the element with the highest priority should be Passenger3. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Verify that the extractMax method of the PriorityQueue class works correctly, successfully removing and returning the Passenger object with the highest priority in the queue. | | | | |
| **Class** | **Method** | **Stage** | **Input values** | **Expected result** |
| PriorityQueue | extractMax | setup2 | passenger1  passenger2  passenger3 | Initially, when attempting to extract the passenger with the highest priority, it should remove Passenger1 and return it. After this, when attempting to extract the passenger with the highest priority, it should remove Passenger2 and return it. Finally, when attempting to extract the passenger with the highest priority, it should remove Passenger3 and return it. |
| PriorityQueue | extractMax | setup1 | None | As the Priority Queue is empty, when attempting to extract the element with the highest priority, it should throw a Priority Queue Exception. |
| PriorityQueue | extractMax | setup3 | passenger3  passenger4  passenger1  passenger5  passenger2 | First, when extracting the Passenger with the highest priority, it should remove Passenger3 and then Passenger4, although both have the same priority it returns the passenger who entered first. After this, when trying to extract the Passenger with the highest priority, it should remove Passenger1 and return it. Then, it extracts Passenger5 and returns it. Finally, when trying to extract the Passenger with the highest priority, it should remove Passenger2 and return it. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Verify that the increaseKey method of the PriorityQueue class works correctly, increasing the key of an element. | | | | |
| **Class** | **Method** | **Stage** | **Input values** | **Expected result** |
| PriorityQueue | increaseKey | setup2 | index = 1, key = 57 | Initially, the element with the highest priority is Passenger 1, but after incrementing the key of Passenger3, when extracting the element with the highest priority, it should remove and return Passenger 3. Later, when extracting the element with the highest priority, it should remove and return Passenger 1. Finally, when extracting the element with the highest priority, it should remove and return Passenger 2. |
| PriorityQueue | increaseKey | setup1 | passenger1, key = 42  index = 0, key 23 | Initially, the key of Passenger1 is 42, but when trying to increment its key to 23, a Priority Queue Exception should be thrown because the new key is smaller than the current one. |
| PriorityQueue | increaseKey | setup1 | key = 42, passenger1  index = 1, key = 56 | As an attempt is made to increase the key of an element at an invalid position in the Priority Queue, as there is only one element, an Index Out Of Bounds Exception will be thrown. |