Diseño de Casos de Prueba

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | Method | Scenario | Input | Expected result |
| HashTable | Insert | setUp1 | We insert a null key and a null task. | If the insertion of null values is not allowed, and the implementation returns false in that case, then the assertEquals (false, flag) assertion will pass. Otherwise, the assertion will fail. |
| HashTable | Insert | setUp1 | We insert a key and task with data. | If the insertion is different from null, then there are no problems. |
| HashTable | Insert | setUp1 | We insert a null key and a null task. | If in the insertion the key is different from null but the values is null then then the test succeed |
| HashTable | Insert | setUp1 | We insert two tasks each with their respective keys. | If the insertion is different from null, then there are no problems. |
| HashTable | Search | setUp2 | We insert the key of the task that we desire. | If the search is correct, then the test succeeds |
| HashTable | Search | setUp2 | We insert the key of the task that we desire. | If the search is null, then the test succeeds |
| HashTable | Delete | setUp2 | We insert the key of the task that we desire. | If the task was deleted correctly then the test succeeds |
| HashTable | Delete | setUp2 | We insert the key of the task that we desire. | If there was a problem with the delete, then the test succeeds. |
| Heap | insert | setUp1 | We insert a null task with a key | Since we inserted a null task then the insertion method returns false, thus since the assertEquals expects a false the task succeeds. |
| Heap | Insert | setUp2 | We insert a task with its respective key | If the insertion of the task was done correctly then the test succeeds. |
| Heap | remove | setUp2 | We enter the task we desire to remove | If the task is removed correctly then the test succeeds. |
| Heap | remove | setUp2 | We enter the task we desire to remove | If the method was not able to remove the task, then it returns false, in that case the test succeeds. |
| Heap | Sort | setUp2 | We enter the method heap.extractMax so that it returns task with the highest priority | If the task returned by the heap.extractMax method is different from the first position of the heap then the test succeeds. |
| Heap | Sort | setUp2 | We insert the method heap.isEmpty | If the Heap is not empty, then the test succeeds. |
| Queue | Insert | setUp1 | We insert an empty queue | If the queue is empty, then the test succeeds. |
| Queue | Peek | setUp2 | We insert the front task. | If the task that was inserted first is returned with the peak, then the task succeeds. |
| Queue | Dequeue | setUp2 | We dequeue three times | If the task, we dequeue matches the want we need then the test succeeds. |
| Queue | Size | setUp2 | We insert the size | If at the beginning, we check if the size is 3 and then after three dequeues the size is 0 then the test succeeds. |
| Queue | IsEmpty | setUp1  setUp2 | We give it two queues one that is empty and another one with tasks | If the size in the beginning is 3 and then zero, then the test succeeds. |
| Queue | Search | setUp3 | We insert five tasks | If the tasks are within the queue, then the test succeeds. |
| Stack | Push | setUp1 | We insert the size of the stack | If the size is 0 and the isEmpty method is true, then the test succeeds. |
| Stack | Pop | setUp2 | We insert the method pop which gives us an action and eliminates the top node. | If at the beginning the size is 3 and after popping one node the size is 2, the test succeeds. |
| Stack | Top | setUp2 | We insert an action. | If the action of the top action is add then the test succeeds. |
| Stack | Size | setUp2 | The input is the size. | If the size at the beginning is three and after a pop the size is two, then the test succeeds. |