**Queue**

|  |  |
| --- | --- |
| **Scenery** | **Resume** |
| setUpScenary1 | A new Queue of Integers with the size of 100 |
| setUpScenary2 | A new Queue of Integers with the size of 1000 |

Test: The method is empty from the queue

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Method** | **Scenery** | **Result** |
| Queue | testIsEmpty1 | setUpScenary1 | True, because the queue has not element in it |
| Queue | testIsEmpty2 | setUpScenary1 | False, because an element has been inserted into the queue |
| Queue | testIsEmpty3 |  | False, the queue is completely full |

Test: Test the enqueue method

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Method** | **Scenery** | **Result** |
| Queue | testEnqueue1 | setUpScenary1 | True, the queue is full |
| Queue | testEnqueue2 | setUpScenary2 | True, the queue is full |

Test: Test the dequeue method

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Method** | **Scenery** | **Result** |
| Queue | testDequeue1 | setUpScenary1 | The size of the queue is decreased by one |
| Queue | testDequeue1 | setUpScenary2 | The size of the queue is decreased by one |

Test: Test the front method

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Method** | **Scenery** | **Result** |
| Queue | testFront1 | setUpScenary1 | The size of the queue is decreased by one and the element retrieved is the first |
| Queue | testFront2 | setUpScenary2 | The size of the queue is decreased by one and the element retrieved is the first |

Test: Test the get sizemethod

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Method** | **Scenery** | **Result** |
| Queue | testGetSize1 | setUpScenary1 | The size of the queue which has items in it if is empty prints 0 |
| Queue | testGetSize2 | setUpScenary2 | The size of the queue which has items in it if is empty prints 0 |

Test: Test the is full

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Method** | **Scenery** | **Result** |
| Queue | testIsFull1 | setUpScenary1 | True the queue is completely full |
| Queue | testIsFull2 | setUpScenary2 | True the queue is completely full |
| Queue | testIsFull3 | setUpScenary1 | False the queue is empty |
| Queue | testIsFull4 | setUpScenary2 | False the queue is empty |

Test: Test the get sizemethod

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Method** | **Scenery** | **Result** |
| Queue | testGetIndexOf1 | setUpScenary1 | The item if the given number as index is returned |
| Queue | testGetIndexOf2 | setUpScenary2 | The item if the given number as index is returned |

**Stack**

|  |  |
| --- | --- |
| **Scenery** | **Resume** |
| setUpScenary1 | A new Stack of Integers with the size of 100 |
| setUpScenary2 | A new Stack of Integers with the size of 1000 |

Test: The method is empty from the stack class

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Method** | **Scenery** | **Result** |
| Stack | testIsEmpty1 | setUpScenary1 | True, because the stack has not element in it |
| Stack | testIsEmpty2 | setUpScenary1 | False, because an element has been inserted into the Stack |
| Stack | testIsEmpty3 | setUpScenary1 | False, the Stack iis completely full |

Test: The will test the insertion and deletions of items

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Method** | **Scenery** | **Result** |
| Stack | testPop1 | setUpScenary1 | Retrieves and removes the last element in the table |
| Stack | testPop2 | setUpScenary2 | Retrieves and removes the last element in the table |
| Stack | testPush1 | setUpScenary1 | Fills the Stack and then shows the elements are successfully inserted |
| Stack | testPush2 | setUpScenary2 | Fills the Stack and then shows the elements are successfully inserted |
| Stack | testTop1 | setUpScenary1 | Retrieves without remove the last element in the table |
| Stack | testTop2 | setUpScenary2 | Retrieves without remove the last element in the table |

Test: The method isfull from the stack class

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Method** | **Scenery** | **Result** |
| Stack | testIfFull1 | setUpScenary1 | The Stack is full then returns true |
| Stack | testIfFull2 | setUpScenary2 | The Stack is full then returns true |
| Stack | testIfFull3 | setUpScenario 1 | The Stack is not full then returns false |
| Stack | testIfFull4 | setUpScenary2 | The Stack is not full then returns false |

**HashTable**

|  |  |
| --- | --- |
| **Scenery** | **Resume** |
| setUpScenary1 | A new hashtable array structure made of integers to 100 |
| setUpScenary2 | A new hashtable array structure made of integers to 1000 |

Test:Test the insert and search operation from hashtable

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Method** | **Scenery** | **Result** |
| HashTable | testInsertTable1() | setUpScenary1 | The element is successfully inserted into the hash\_table hence returns True |
| HashTable | testInsertTable2() | setUpScenary2 | The element is sucessfully inserted into the hashtable hence returns True |
| HashTable | searchTable1() | setUpScenary1 | The key is right an corresponds to an element in the hash table therefore return True |
| HashTable | searchTable1() | setUpScenary2 | The given key is right an corresponds an item in the hash, so the hash function works and goes to the index where is located the item |