**DESIGN OF THE SCENARIOS:**

| **Class** | **Name** | **Scenario** |
| --- | --- | --- |
| HasTableTest | setup stage 1 | The Hashtable is empty |
| QueueTest | setup Stage1 | The Queue is empty |
| StackTest | setup Stage1 | The Stack is empty |
| setup Stage2 | The Stack has elements |
| PriorityQueueTest | setup Stage 1 | The PriorityQueue is empty |
| setup Stage 2 | The PriorityQueue has elements |
| HeapTest | setup Stage 1 | The Heap has elements |
| ActivityManagerTest | setupStage1 | The data structures are empty |
| setupStage2 | The data structures have activities |
| ControllerTest | setupStage1 | The stack is empty, and the ActivityManager is empty |
| setupStage2 | The stack is empty, and the ActivityManager has activities |

**TEST CASE DESIGN:**

**HashTable:**

| **Test Objective:** Verify the functionality of the put method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| HashTableTest | testPutAndGet | setupStage1 | “Key”  “42” | Expected retrieved value to be 42 |
| testPutAndGetMultiple | setupStage1 | Key1: "key1", Value1: 42,  Key2: "key2",  Value2: 123 | Expected retrievedValue1 to be 42 and retrievedValue2 to be 123. |

| **Test Objective:** Verify the functionality of the remove method in the hashtable. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| HashTableTest | testRemove | setupStage1 | Key: "key1" | Expected contains("key1") to be false and get("key1") to be null after removing key "key1". |
| testRemoveNonExistentKey | setupStage1 | Key: "nonExistentKey" | Expected contains("nonExistentKey") to be false and get("nonExistentKey") to be null after attempting to remove a non-existent key. |

| **Test Objective: :** Verify the functionality of the contains method in the hashtable. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| HashTableTest | testContains | setupStage1 | Key: "key", Value: 42 | Expected contains("key") to be true after adding key "key" and value 42. |
| testContainsNonExistentKey | setupStage1 | Key: "nonExistentKey" | Expected contains("nonExistentKey") to be false as this key has not been added previously. |

| **Test Objective: :**  Verify the functionality of the isEmpty method in the hashtable. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| HashTableTest | testIsEmpty | setupStage1 |  | Expected the hashtable to be empty, so isEmpty() should return true. |
| testIsEmptyAfterInsert | setupStage1 | Key: "key", Value: 42 | Expected that after adding a key-value pair, the hashtable is no longer empty, so isEmpty() should return false |

| **Test Objective: :**  Verify the functionality of the size method in the hashtable. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| HashTableTest | testSize | setupStage1 | Key1: "key1", Value1: 42, Key2: "key2", Value2: 123 | Expected that after adding two key-value pairs, the size of the hashtable is 2. |
| testSizeAfterInsertAndRemove | setupStage1 | Key1: "key1", Value1: 42, Key2: "key2", Value2: 123 | Expected that after adding two key-value pairs and removing one of them, the size of the hashtable is 1. |

**Queue:**

| **Test Objective:** Verify the functionality of the enqueue method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| QueueTest | testEnqueueOneElement | setupStage1 | “item” | Not exception, the queue is not empty; it contains "item1." |
| testEnqueueTwoElements | setupStage1 | “item1”, “item2” | The queue is not empty; it contains "item1" and "item2". |
| testEnqueueOneElementGetsOnFront | setupStage1 | “item1”, “item2” | The queue is not empty; the front is "item1," not "item2." |

| **Test Objective:** Verify the functionality of the front method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| QueueTest | testFrontEmptyQueue | setupStage1 |  | Exception is thrown |
| testFrontNonEmpty | setupStage1 | "item1" | The front of the queue is "item2." |
| testFrontAfterDequeue | setupStage1 | "item1" | Exception is thrown |

| **Test Objective:** Verify the functionality of the dequeue method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| QueueTest | testDequeueWhenCreated | setupStage1 |  | The result is null. |
| testDequeueWhenOneElements | setupStage1 | "item1" | The result is "item1," and the queue is empty. |
| testDequeueWhenMultipleElements | setupStage1 | "item1", "item2", "item3" | The result is "item2," and the queue is not empty. |

| **Test Objective:** Verify the functionality of the dequeue method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| QueueTest | testIsEmptyAfterCreation | setupStage1 |  | The queue is empty. |
| testIsEmptyAfterEnqueue | setupStage1 | "item1" | The queue is not empty. |
| testIsEmptyAfterDequeueMultipleElements | setupStage1 | "item1", "item2" | The queue is empty. |

**Stack:**

| **Test Objective:** Verify the functionality of the size method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| StackTest | testSizeEmpty | setupStage1 |  | the size of the stack is 0 |
| testSizeFilled | setupStage2 |  | The size of the stack is different from 0 |

| **Test Objective:** Verify the functionality of the isEmpty method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| StackTest | testIsEmptyeEmpty | setupStage1 |  | true |
| testIsEmptyFilled | setupStage2 |  | false |

| **Test Objective:** Verify the functionality of the push method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| StackTest | testPushEmpty | setupStage1 | “item” | No exceptions, size = 1 and top = “item” |
| testPushFilled | setupStage2 | “item” | No exceptions, size increases by 1 and top = “item” |
| testPushNull | setupStage1 | null | exception is thrown |

| **Test Objective:** Verify the functionality of the top method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| StackTest | testTopEmpty | setupStage1 |  | Exception is thrown |
| testTopFilled | setupStage2 |  | No exception, it must return the last pushed element |

| **Test Objective:** Verify the functionality of the pop method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| StackTest | testPopEmpty | setupStage1 |  | Exception is thrown |
| testPopFilled | setupStage2 |  | No exception, it must return the removed item and decrease size by 1. |

**ActivityManager**

| **Test Objective:** Test the addActivity method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| ActivityManagerTest | testAddPriorActivityEmpty | setupStage1 | Activity with priority | Activity added on hash table and PriorityQueue |
| testAddPriorActivityNotEmpty | setupStage2 | Activity with the highest priority | Activity added on hash table and at the root of the PriorityQueue |
| testAddNonPriorActivityEmpty | setupStage1 | Activity with no priority | Activity added on hash table and at the front of the Queue |
| testAddNonPriorActivityNotEmpty | setupStage2 | Activity with no priority | Activity added on hash table and at the back of the Queue |

| **Test Objective:** Test the getActivity method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| ActivityManagerTest | testGetActivityNotExists | setupStage1 | Non existent title | Null |
| testGetActivityExists | setupStage2 | Existent title | Corresponding activity |

| **Test Objective:** Test the containsActivity method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| ActivityManagerTest | testContainsActivityNonExists | setupStage1 | Non existent title | false |
| testContainsActivityExists | setupStage2 | Existent title | true |

| **Test Objective:** Test modifyActivity method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| ActivityManagerTest | testModifyActivityFound | setupStage2 | Original Activity,  Updated Activity | The original activity should be successfully replaced with the updated activity, and the attributes of the modified activity should match the updated values. No exceptions should be thrown. |
| testModifyActivityNotFound | setupStage2 | Updated Activity | QueueException with "The activity does not exist." should be thrown when attempting to modify a non-existent activity. |

| **Test Objective:** Verify the functionality of the removeActivity method to remove activities from the Activity Manager. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| ActivityManagerTest | testRemoveActivityFromEmptyManager | setupStage1 | Activity "task" | The "task" activity has been successfully removed from the manager. The "task" activity is no longer in the manager. |
| testRemoveActivityThatDoesNotExist | setupStage1 | Existing Activity "existing" Non-Existing Activity "non-existing" | The "non-existing" activity is not in the manager. The "existing" activity is still in the manager. |
| testRemoveActivityFromManager | setupStage1 | Activity "task" | The "task" activity has been successfully removed from the manager. The "task" activity is no longer in the manager. |

**Controller**

| **Test Objective:** Test the registerActivity method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| ControllerTest | testRegisterActivitySuccess | setUpStage1 | Valid input values | "The activity was registered successfully" |
| testRegisterActivityAlreadyExists | setUptStage2 | existing title | "The activity already exists" |

| **Test Objective:** Test modifyActivity method | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| ControllerTest | testModifyActivitySuccess | setUpStage1 | "task" ,"description" 1, 1    "task","newdescription", 2 ,2 | The activity was modified successfully. The description of the activity should change to "new description," its priority should be set to non-priority, and its type should be a reminder. |
| testModifyActivityNotFound | setUpStage1 | "non\_existent\_task" , "new description", 2 ,2 | Activity not found. Attempting to modify a non-existent activity should return an error. |

| **Test Objective:** Test the undoLastAction method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| ControllerTest | testUndoLastActionEmpty | setUpStage1 |  | "There is no action to undo" |
| testUndoLastActionAdd | setUptStage1 | Last action was ADD | "The add activity action was undone successfully" |
| testUndoLastActionRemove | setUptStage1 | Last action was REMOVE | "The remove activity action was undone successfully" |
| testUndoLastActionModify | setUptStage1 | Last action was MODIFY | "The last action was undone successfully" |
| testUndoMultipleActions | setUpStage2 | Mix of the three actions | success message and the stack should be empty |

| **Test Objective: V**erify the functionality of the removeActivity method to remove activities from the Activity Manager. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| ControllerTest | testRemoveActivitySuccess | setUpStage1 | Activity "task3" | "The activity was removed successfully" is returned. |
| testRemoveActivityNotFound | setUpStage1 | Activity "non-existing | "Activity not found" is returned. |

**PriorityQueue**

| **Test Objective: Verify the functionality of the toString method.** | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| HeapTest | testToString | setUpStage1 | 12, 1, 8 | "12 1 8 " |
| testToStringWithEmptyHeap | setUpStage1 |  | “” |

| **Test Objective: Verify the functionality of the buldHeap method** | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| HeapTest | testBuildHeap | setupStage1 | 10, 5, 7 | "10 5 7 " |
| testBuildHeap | setupStage2 | 15, 2, 20 | "20 2 15 " |
| testBuildHeapWithDuplicateValues | setupStage1 | 5, 5, 5, 10, 10, 3 | "10 10 5 5 5 3 " |
| testBuildHeapWithDuplicateValues | setupStage2 | 20, 15, 25, 10, 5 | "25 15 20 10 5 " |

| **Test Objective: Verify the functionality of the heapSort method** | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| HeapTest | testHeapSort | setUpStage1 | 15, 3, 10 | "3 10 15 " |
| testHeapSort | setUpStage1 | 50, 30, 20 | "20 30 50 " |

**PriorityQueue**

| Test Objective: Test the insert and extractRoot methods. | | | | |
| --- | --- | --- | --- | --- |
| Class | Method | Scenario | Input Values | Result |
| PriorityQueueTest | testInsertAndExtractRoot | setupStage1 | 5, 7, 3, 1 | The extracted root should be 7 |
| testInsertAndExtractMultiple | setupStage2 | , 7, 3, 1 | The extracted values should be in the order of 7, 5, 3, and 1 |

| Test Objective: Test the getRoot and extractRoot methods. | | | | |
| --- | --- | --- | --- | --- |
| Class | Method | Scenario | Input Values | Result |
| PriorityQueueTest | testGetRootWhenEmpty | setupStage1 | – | The result should be null |
| testExtractRootWhenEmpty | setupStage2 | – | A QueueException with “The queue is empty.” should be thrown. |

| **Test Objective:** Test the setKey method. | | | | |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Scenario** | **Input Values** | **Result** |
| PriorityQueueTest | testSetKey | setupStage1 | 3, 5, 2; index=0, newKey=5. | The extracted values should be in the order of 5, 3, and 2 |