**HashTableTest Unit Case Design**

|  |  |  |
| --- | --- | --- |
| **Name** | **Class** | **Scenary** |
| setupScenary1 | HashTable | 3 estanterías vacías, creadas, cada una con distinto tamaño de estantería (2 para la primera estantería, 3 para la segunda estantería y 5 para la tercera). |
| setupScenary1 | HashTable | La primera estantería cuenta con dos juegos: game1(code:8765, amount:1, price: 34000) y game2(code: 32145, amount: 3, price:28000). |
| setupScenary1 | HashTable | La segunda estantería cuenta con tres juegos: game3(code: 6753, amount: 2, price: 73000), game4(code: 8900, amount:6, price: 50000) y game5(1234, 1, 90000). |
| setupScenary1 | HashTable | La tercera estantería cuenta con cinco juegos: game1(code:8765, amount:1, price: 34000), game2(code: 32145, amount: 3, price:28000), game3(code: 6753, amount: 2, price: 73000), game4(code: 8900, amount:6, price: 50000) y game5(1234, 1, 90000). |

**HashTableTest**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Validate that a shelf is correctly entered into the HashTable | | | | |
| **Class** | **Method** | **Scenary** | **Input Values** | **Result** |
| HashTableTest | addTest() | setUpScenary1 | Shelfs 1, 2 and 3 previously created | The shelves were correctly entered into the HashTable. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Validate that a HashTable shelf is correctly searched | | | | |
| **Class** | **Method** | **Scenary** | **Input Values** | **Result** |
| HashTableTest | searchTest() | setUpScenary1 | Shelfs 1, 2 and 3 previously created | A shelf of the HashTable was found correctly. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Validate that a non-existent shelf is not found in the HashTable | | | | |
| **Class** | **Method** | **Scenary** | **Input Values** | **Result** |
| HashTableTest | searchTest2() | setUpScenary1 | Ninguno | Null, the shelf was not found because it was not previously entered. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Validate the correct elimination of a shelf in the HashTable | | | | |
| **Class** | **Method** | **Scenary** | **Input Values** | **Result** |
| HashTableTest | deleteAndSearchTest() | setUpScenary1 | Shelfs 1, 2 and 3 previously created | Null, the shelf was successfully cleared from the HashTable. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Validate that the indexes where the shelves were added are different in each case | | | | |
| **Class** | **Method** | **Scenary** | **Input Values** | **Result** |
| HashTableTest | getIndexTest() | setUpScenary1 | Shelfs 1, 2 and 3 previously created | The indexes where the shelves were added are different from each other. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Validate that two shelves with the same Key are not added to the HashTable | | | | |
| **Class** | **Method** | **Scenary** | **Input Values** | **Result** |
| HashTableTest | duplicatedKeyTest() | setUpScenary1 | Shelfs 1, 2 and 3 previously created | There are no shelves with the same Key in the HashTable |

**QueueTest Unit Case Design**

|  |  |  |
| --- | --- | --- |
| **Name** | **Class** | **Scenary** |
| setupScenary1 | Queue | 5 clients added with different codes, each one and an empty game list. |
| setupScenary1 | Queue | The first client with code: "pdr34" and list of games: "games" |
| setupScenary1 | Queue | The second client with code: "sdio29n" and list of games: "games" |
| setupScenary1 | Queue | The third client with code: "opsdf893" and games list: "games" |
| setupScenary1 | Queue | The fourth client with code: "op6op2" and list of games: "games" |
| setupScenary1 | Queue | The fifth client with code: "mds3j" and list of games: "games" |

**QueueTest Test Case Design**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Validate that a customer is correctly added to the Cake and remains in front | | | | |
| **Class** | **Method** | **Scenary** | **Input Values** | **Result** |
| QueueTest | addTest() | setUpScenary1 | Clients 1, 2, 3, 4, and 5 previously created | Clients have been added successfully and remains in front |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Validate that when polling returns and removes a client from the Queue | | | | |
| **Class** | **Method** | **Scenary** | **Input Values** | **Result** |
| QueueTest | pollTest() | setUpScenary1 | Clients 1, 2, 3, 4, and 5 previously created | The client is returned and removed from the Queue in a successful way |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Validate that when polling returns a client from the Queue | | | | |
| **Class** | **Method** | **Scenary** | **Input Values** | **Result** |
| QueueTest | peekTest() | setUpScenary1 | Clients 1, 2, 3, 4, and 5 previously created | The client is returned from the Queue in a successful way |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Validates that clients are correctly added and the current size of the Queue is returned at all times | | | | |
| **Class** | **Method** | **Scenary** | **Input Values** | **Result** |
| QueueTest | sizeTest() | setUpScenary1 | Clients 1, 2, 3, 4, and 5 previously created | The client is added correctly and the current size of the Queue is returned at all times |

**StackTest Unit Case Design**

|  |  |  |
| --- | --- | --- |
| **Name** | **Class** | **Scenary** |
| setupScenary1 | Stack | 5 empty games, created, with different codes, each one, a quantity, and a price |
| setupScenary1 | Stack | The first game has a code: “8765”, quantity: 1 and price: 34,000 |
| setupScenary1 | Stack | The second game has a code: "32145", quantity: 3 and price: 28,000 |
| setupScenary1 | Stack | The third game has a code: “6753”, quantity: 2 and price: 73,000 |
| setupScenary1 | Stack | The fourth game has a code: "6753", quantity: 6 and price: 50,000 |
| setupScenary1 | Stack | The fourth game has a code: "6753", quantity: 6 and price: 50,000 |

**StackTest Case Design**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Validate that a game is correctly added to the Stack | | | | |
| **Class** | **Method** | **Scenary** | **Input Values** | **Result** |
| StackTest | pushAndPopNullTest () | setUpScenary1 | Games 1, 2, 3, 4, and 5 are previously created | A game was successfully added to the Stack. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Validate that the last game that was entered into the Stack returns correctly | | | | |
| **Class** | **Method** | **Scenary** | **Input Values** | **Result** |
| StackTest | popTest () | setUpScenary1 | Games 1, 2, 3, 4, and 5 are previously created | The last game that was entered into the Stack was correctly returned. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Validate the correct elimination of a game in the Stack | | | | |
| **Class** | **Method** | **Scenary** | **Input Values** | **Result** |
| StackTest | peekAndPopTest () | setUpScenary1 | Games 1, 2, 3, 4, and 5 are previously created | Null, the game was correctly returned and removed from the stack. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test objective:** Check that the last added element is at the end of the Stack | | | | |
| **Class** | **Method** | **Scenary** | **Input Values** | **Result** |
| StackTest | compareTopTest() | setUpScenary1 | Games 1, 2, 3, 4, and 5 are previously created | The last item added is at the end of the Stack. |