**Formato de escenarios y casos de uso**

Integrantes:

Leidy Daniela Londoño - A00392917

Isabella Huila Cerón - A00394751

TotalPruebas: 26

**Configuración de los Escenarios**

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Clase** | **Escenario** |
| setupStage1 | InventoryTest | Un objeto de la clase Inventory con un Objeto de la clase Product:   * name= “Producto A”, descripción= “xxxxx A”, categoría= Books, precio= 10000, cantidad= 6 |
| setupStage2 | InventoryTest | Un objeto de la clase Inventory con dos objetos de la clase Product:   * name= “Producto B”, descripción= “xxxxx B”, categoría= Plays, precio= 20000, cantidad= 2 * name= “Producto C”, descripción= “xxxxx C”, categoría= Drinks, precio= 6000, cantidad 2 |
| setupStange3 | ProductTest | Un objeto de la clase Product  name= “Producto A”, descripción= “xxxxx A”, categoría= Books, precio= 10000, cantidad= 6 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método addProduct de la clase Invetory agregue correctamente un nuevo producto tanto cuando el producto es nuevo (no existe un producto con dicho nombre) y verificar que cuando se cree el producto y se vaya agregar al inventario, no tenga en el precio valores negativos y en la cantidad. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado** |
| Product | createProduct | setupStage3 | name= “Producto D”, descripción= “xxxx D”, categoría= Clothe, precio= -50000, cantidad= 2 | Lanza Exception y el mensaje dice que no puede usar números negativos |
| Product | createProduct | setupStage3 | name= “Producto E”, descripción= “xxxx E”, categoría= Clothe, precio=50000, cantidad= -3 | Lanza Exception y el mensaje dice que no puede usar números negativos |
| Inventory | addProduct | setupStage1 | name= “Producto F”, descripción= “xxxx F”, categoría= Food, precio=53000, cantidad= 8 | El inventario queda ahora con dos productos, el primero con nombre “Producto A” y el segundo “Producto F” |
| Inventory | addProduct | setupStage1 | name= “Producto A”, descripción= “xxxxx A”, categoría= Books, precio= 10000, cantidad= 6 | El inventario queda solo con el producto que tenía. Lanza una Exception y el mensaje dice que el producto ya existe en el inventario |
| Inventory | removeProduct | setupStage2 | name= “Producto C” | El inventario queda con un producto que tiene el nombre “Producto B” |
| Inventory | removeProduct | setupStange2 | Name= “Producto H” | El inventario queda con los dos productos que tenía. Lanza una Exception y el mensaje dice que está tratando de eliminar un producto que no existe en el inventario |

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Clase** | **Escenario** |
| setupStage1 | OrderTest | Un objeto de la clase Order, que contiene una lista de productos   * name= “Comprador A”, productsList= products, total= 2000, fecha= 16/04/23   La lista de productos contiene=   * name= “Producto A”, descripción= “xxxxx A”, categoría= Books, precio= 1000, cantidad= 1 * name= “Producto B”, descripción= “xxxxx B”, categoría= food, precio= 1000, cantidad= 2 |
| setUpStage2 | OrdersTest | Un objeto de la clase Orders con un objeto de la clase Order   * name= “Producto B”, descripción= “xxxxx B”, categoría= food, precio= 1000, cantidad= 2 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que una orden se cree correctamente, con los atributos establecidos y que se puedan agregar productos a la orden. | | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado** |
| Order | orderConstructor | setupStage1 | name= “Comprador A”, productsList= products, total= 2000, fecha= 16/04/23 | La Orden se crea correctamente con los atributos establecidos |
| Order | addProductToOrder | setupStage1 | name= “Producto C”, descripción= “xxxxx A”, categoría= Drink, precio= 2000, cantidad= 1 | La orden recibe un nuevo producto, el cual tiene nombre “Product C” |
| Orders | addOrder | setupStage2 | name= “Comprador A”, productsList= products, total= 2000, fecha= 16/04/23 | La lista de ordenes ahora va a tener dos órdenes una con el nombre del comprador “Comprador B” y en la otra con “Comprador A” |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que los métodos increaseProduct y decreaseProduct de la clase Product funcionen correctamente, adicionando la cantidad pasado por parámetro la cantidad actual del Producto. | | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado** |
| Product | increaseProduct | setupStage1 | Cantidad = 3 | La cantidad del producto aumenta en 3 a la cantidad actual que se tenía |
| Product | increaseProduct | setupStage1 | Cantidad = -10 | La cantidad del producto queda igual. Lanza una excepción ya que no se pueden colocar cantidades negativas |
| Product | decreaseProduct | setupStage1 | Cantidad = 4 | La cantidad del producto disminuye tres veces su cantidad, a la actual |
| Product | decreaseProduct | setupStage1 | Cantidad = -1 | La cantidad del producto queda igual. Lanza una excepción ya que no se pueden colocar valores negativos. |

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Clase** | **Escenario** |
| setupStage2 | ProductTest | Un objeto de la clase Inventory con cinco objetos de la clase Product:   * name= “Producto B”, descripción= “xxxxx B”, categoría= Plays, precio= 20000, cantidad= 2 * name= “Producto C”, descripción= “xxxxx C”, categoría= Drinks, precio= 6000, cantidad 2 * name= “Producto A”, descripción= “xxxxx A”, categoría= Books, precio= 20000, cantidad= 6 * name= “Producto D”, descripción= “xxxxx D”, categoría= Drinks, precio= 6000, cantidad 5 * name= “Producto E”, descripción= “xxxxx E”, categoría= Drinks, precio= 2000, cantidad 2 |
| setupStage1 | OrderTest | Un objeto de la clase Orders con tres objeto de la clase Order   * name= “Comprador A”, productsList= products, total= 2000, fecha= 16/04/23 * name= “Comprador B”, productsList= products, total= 55000, fecha= 6/04/23 * name= “Comprador C”, productsList= products, total= 48000, fecha= 9/04/23 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que se pueda buscar un producto y un pedido. En el caso del producto, al ingresar como parámetro el nombre, precio, categoría o número de veces comprado, el método debería devolver los productos. En el caso del pedido, al ingresar como parámetro el nombre de comprador, precio total o fecha de compra, el método debería devolver los pedidos. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| Inventory | searchProduct | setupStage2 | name= “Producto C” | Se devuelve la información del producto con este nombre |
| Inventory | searchProduct | setupStage2 | precio= 20000 | se devuelve la información de los dos productos con este precio, el primero con el nombre de “Producto B” y el otro “Producto A” |
| inventory | searchProduct | setupStage2 | Categoría = Drinks | Se devuelve la información de los tres productos con la categoría del producto. El primero con el nombre “Producto C”, el segundo “Producto D ” y “Producto E” |
| Inventory | searchProduct | setupStage2 | Número de veces comprado = 5 | Se devuelve la información del producto con respecto al número de veces comparado con el nombre “Producto C” |
| Inventory | searchProduct | setupStage2 | name= “Producto H” | no se encuentra ningún producto con este nombre |
| Orders | searhOrder | setupStage1 | nameBuyer= “Comprador A” | Se devuelve la información de la orden que tiene como comprador A |
| Orders | searchOrder | setupStage1 | precio total = 2000 | Se devuelve la información del Orden con base a la precio total del “Comprador A” |
| Orders | searchOrder | setupStage1 | fechaCompra= 9/04/23 | Se devuelve la información de la orden que tiene esta fecha |
| Orders | searchOrder | setupStage1 | nameBuyer= “Comprador H” | No se encuentra ninguna orden con este nombre |

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Clase** | **Escenario** |
| setupStage3 | InventoryTest | Un objeto de la clase Inventory con cinco objetos de la clase Product:   * name= “Calabozos y dragones”, descripción= “xxxxx B”, categoría= Plays, precio= 20000, cantidad= 10, numeroVeces= 3 * name= “Coco-Lemonade”, descripción= “xxxxx C”, categoría= Drinks, precio= 6000, cantidad 8, numerVeces= 4 * name= “El diario de Ana Frank”, descripción= “xxxxx A”, categoría= Book, precio= 30000, cantidad= 6, numeroVeces= 2 * name= “Agua”, descripción= “xxxxx D”, categoría= Drinks, precio= 1000, cantidad 5, numerVeces= 3 * name= “Durazno-Té”, descripción= “xxxxx E”, categoría= Drinks, precio= 2000, cantidad 2, numerVeces= 1 |
| setupStage1 | OrdersTest | Un objeto de la clase Orders con cinco objetos de la clase Order   * name= “Azul”, productsList= products, total= 2000, fecha= 16/04/23 * name= “Betty”, productsList= products, total= 55000, fecha= 6/04/23 * name= “Homero”, productsList= products, total= 48000, fecha= 9/04/23 * name= “Daniela”, productsList= products, total= 10000, fecha= 1/04/23 * name= “Carmen”, productsList= products, total= 39000, fecha= 7/04/23 |

**Diseño de Casos de Prueba**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que en la búsqueda por rango devuelva los productos y orden dependiendo del parámetro de entrada. Además, que este en el orden (ascendente o descendente) que se haya pedido | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| Inventory | searchRangeProduct | setupStage2 | orden= Descendente  valorMinimo= 6000 y valorMaximo= 30000 | Se devuelve los productos que están entre este rango y de forma descendente. En este caso sería:   * El diario de Ana Frank” * Calabozos y dragones * Coco-Lemonade |
| Inventary | searchRangeProduct | setupStage2 | letra de inicio = “A”  Letra final = “D”  Orden = ascendente | Se devuelve la información de los productos que estén entre ese rango y de forma Ascendente de los prefijos entre “A” y “D”. en este caso devolvería:   * Agua * Calabozos y dragones * Coco-Lemonade * Durazno-Té |
| Orders | searchRangeOrder | setupStage1 | valorMinimo= 10000  valorMaximo= 50000  Orden= Ascendete | Se devuelve los productos que están entre este rango. En este caso sería:   * “Daniela” * “Carmen” * “Homero” |
| Orders | searchRangeOrder | setupStage1 | Letra de inicio = “A”  Letra final = “D”  orden = Descendente | Se devuelve la información de los productos que estén entre ese rango de los compradores entre “A” y “D”. en este caso devolvería:   * Daniela * Carmen * Betty * Azul |