**Formato de escenarios y casos de uso**

**Configuración de los Escenarios**

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Clase** | **Escenario** |
| setUp() | StoreTest | 1. **Usuario 1**:    * **Nombre de usuario**: “user1”    * **Tipo de usuario**: Client (false)    * **Contraseña**: “clave123”    * **Nombre**: “Ana”    * **Apellido**: “García”    * **Fecha de nacimiento**: “1990-05-15”    * **Ciudad**: “Ciudad”    * **Dirección**: “Calle 123,”    * **Teléfono**: “+57 300 123 4567”    * **Correo electrónico**: “ana@example.com”    * **Método de pago**: CreditCard ()   Número de cuenta: “987654321”  Código: “456”  Número de cuotas: 12   1. **Usuario 2**:    * **Nombre de usuario**: “user2”    * **Tipo de usuario**: Seller (true)    * **Contraseña**: “admin123”    * **Nombre**: “Carlos”    * **Apellido**: “Martínez”    * **Fecha de nacimiento**: “1985-09-20”    * **Dirección**: “Av. Principal, Ciudad”    * **Teléfono**: “+57 310 987 6543”    * **Correo electrónico**: “carlos@example.com” 2. **Usuario 3**:    * **Nombre de usuario**: “user3”    * **Tipo de usuario**: Client (false)    * **Contraseña**: “segura456”    * **Nombre**: “María”    * **Apellido**: “López”    * **Ciudad**: “Ciudad”    * **Fecha de nacimiento**: “1998-03-10”    * **Dirección**: “Carrera 456, Ciudad”    * **Teléfono**: “+57 320 555 7890”    * **Correo electrónico**: “maria@example.com”    * **Método de pago**: PSE   Banco: “Banco del Valle”   * + Teléfono : “+57 320 555 7890”   + **Método de pago**: DebitCard   Número de cuenta: “123456789”  Código: “789” |
| setUp1() | StoreTest | **productCode:** “E5678”  **timesPurchased:** 20  **productName:** “Smartphone Galaxy S21”  **productDescription: “**Un teléfono inteligente con pantalla AMOLED y cámara de alta resolución.”  **price:** “$699.99”  **stock:** 30  **productCategory:** Electrónica  **productCode:** “R9876”  **timesPurchased:** 15  **productName:** “Camiseta básica”  **productDescription:** “Una camiseta cómoda y versátil para uso diario.”  **price:** “$19.99”  **stock:** 25  **productCategory:** Ropa y accesorios  **productCode:** “B1234”  **timesPurchased:** 5  **productName: “**El Gran Gatsby”  **productDescription: “**Una novela clásica ambientada en la década de 1920.”  **price:** “$15.99”  **stock:** 10  **productCategory: Libros**  **productCode:** “B1235”  **timesPurchased:** 25  productName: “El principito”  **productDescription: “**Una novela clásica de aventura e introspeccion  **price:** “$13.99”  **stock:** 20  **productCategory:** Libros |
| setUp2() |  | **productID**: 12345  **quantity**: 5  **methodPayment**: Tarjeta de crédito  **date**: 2024-03-10  **productID**: 67890  **quantity**: 3  **methodPayment**: PayPal  **date**: 2024-03-11  **productID**: 54321  **quantity**: 2  **methodPayment**: Transferencia bancaria  **date**: 2024-03-12 |

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Clase** | **Escenario** |
|  |  |  |
|  |  |  |
|  |  |  |

…

\* El nombre de los escenarios puede ser setupStage1, setupStage2, etc.

\* La clase es la clase de testing correspondiente al modelo donde acontece el escenario. Por ejemplo si usted está probando User, clase será UserTest.

\* El escenario es la descripción de las condiciones iniciales del escenario.

**Diseño de Casos de Prueba**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** verificar el correcto funcionamiento de la creación de usuarios, en caso de ser un usuario seller o client; el primer caso corresponde añadir uno tipo seller, el segundo revisar si se impide añadir usuarios tipo seller con el mismo username , el tercero añadir client, el cuarto revisar si no deja registrar usuarios con mismo username y por ultimo si al añadir un usuario tipo client se permite agregar más de un método de pago. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| Store | createUserSeller() | Don´t need | **Nombre de usuario**: “user2”  **Tipo de usuario**: Seller (true)  **Contraseña**: “admin123”  **Nombre**: “Carlos”  **Apellido**: “Martínez”  **Fecha de nacimiento**: “1985-09-20”  **Dirección**: “Av. Principal, Ciudad”  **Teléfono**: “+57 310 987 6543”  **Correo electrónico**: “carlos@example.com” | True. |
| Store | createUserSeller() | Don´t need | **Nombre de usuario**: “user2”  **Tipo de usuario**: Seller (true)  **Contraseña**: “admin123”  **Nombre**: “Carlos”  **Apellido**: “Martínez”  **Fecha de nacimiento**: “1985-09-20”  **Dirección**: “Av. Principal, Ciudad”  **Teléfono**: “+57 310 987 6543”  **Correo electrónico**: “carlos@example.com” | False |
| Store | createUserClient() | Don´t need | **Nombre de usuario**: “user1”  **Tipo de usuario**: Client (false)  **Contraseña**: “clave123”  **Nombre**: “Ana”  **Apellido**: “García”  **Fecha de nacimiento**: “1990-05-15”  **Ciudad**: “Ciudad”  **Dirección**: “Calle 123”  **Teléfono**: “+57 300 123 4567”  **Correo electrónico**: “ana@example.com”  **Método de pago**: CreditCard  -Número de cuenta: “987654321”  -Código: “456”  -Número de cuotas: 12 | true |
| Store | createUserClient() | Don´t need | **Nombre de usuario**: “user1”  **Tipo de usuario**: Client (false)  **Contraseña**: “clave123”  **Nombre**: “Ana”  **Apellido**: “García”  **Fecha de nacimiento**: “1990-05-15”  **Ciudad**: “Ciudad”  **Dirección**: “Calle 123”  **Teléfono**: “+57 300 123 4567”  **Correo electrónico**: “ana@example.com”  **Método de pago**: CreditCard  -Número de cuenta: “987654321”  -Código: “456”  -Número de cuotas: 12 | false |
| Store | createUserClient | Don´t need | **Nombre de usuario**: “user3”  **Tipo de usuario**: Client (false)  **Contraseña**: “segura456”  **Nombre**: “María”  **Apellido**: “López”  **Ciudad**: “Ciudad”  **Fecha de nacimiento**: “1998-03-10”  **Dirección**: “Carrera 456, Ciudad”  **Teléfono**: “+57 320 555 7890”  **Correo electrónico**: “maria@example.com”  **Método de pago**: PSE  -Banco: “Banco del Valle”  -Teléfono : “+57 320 555 7890”  **Método de pago**: DebitCard  -Número de cuenta: “123456789”  -Código: “789” | true |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** verificar el correcto funcionamiento de la función de logIn() en caso de registrarse como Client o Seller, con datos ya existentes en el sistema y en caso de digitar algún nombre de usuario o contraseña erróneamente | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| Store | logIn | setUp() | **Nombre de usuario**: “user1”  **Contraseña**: “clave123” | true |
| **Nombre de usuario**: “user2”  **Contraseña**: “admin123” | true |
| **Nombre de usuario**: “uSer1”  **Contraseña**: “clave123” | false |
| **Nombre de usuario**: “user2”  **Contraseña**: “admn123” | false |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** verificar si la función de añadir inventario funciona correctamente, añadiendo primero de forma normal, el segundo es en caso de cambiar la categoría, y por ultimo en caso de agregar un producto con el mismo código incluso si los demás atributos son diferentes | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| Store | addInventary() | Don´t need | **productCode:** “B1234”  **timesPurchased:** 5  **productName: “**El Gran Gatsby”  **productDescription: “**Una novela clásica ambientada en la década de 1920.”  **price:** “$15.99”  **stock:** 10  **productCategory:** Libros | true |
| **productCode:** “E5678”  **timesPurchased:** 20  **productName:** “Smartphone Galaxy S21”  **productDescription: “**Un teléfono inteligente con pantalla AMOLED y cámara de alta resolución.”  **price:** “$699.99”  **stock:** 30  **productCategory:** Electrónica | true |
| **productCode:** “B1234”  **timesPurchased:** 25  productName: “El principito”  **productDescription: “**Una novela clásica de aventura e introspección”  **price:** $13.99  **stock:** 20  **productCategory:** Libros | false |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** verifica si la función de buscar producto o inventario arroja el objeto esperado, en este caso si fue por nombre, por código del producto, por precio mínimo, por precio máximo o por categoría, o en caso de poner algo que no se encuentra en la base de datos. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| Store | searchInventary | setUp1() | B1235 | **Product**(“B1235”, 5, “El principito”, “Una novela clásica de aventura e introspección”, 13.99, 20, Libros) |
| El principito | **Product**((“B1235”, 5, “El principito”, “Una novela clásica de aventura e introspección”, 13.99, 20, Libros) |
| minPrice : $10 | **Product**((“B1235”, 5, “El principito”, “Una novela clásica de aventura e introspección”, 13.99, int 20, Libros)  **Product**(“R9876”, 15, Camiseta básica,  “Una camiseta cómoda y versátil para uso diario” 19.99, 25, Ropa y accesorios)  **Product**(“B1234”, 5, “El Gran Gatsby”  “Una novela clásica ambientada en la década de 1920”., 15.99, 10, Libro)  **Product**(“E5678”, 20, “Smartphone Galaxy S21”, “Un teléfono inteligente con pantalla AMOLED y cámara de alta resolución.”., 699.99, 30, Electrónica) |
| maxPrice: $20 | **Product**((“B1235”, 5, “El principito”, “Una novela clásica de aventura e introspección”, 13.99, int 20, Libros)  **Product**(“R9876”, 15, Camiseta básica,  “Una camiseta cómoda y versátil para uso diario” 19.99, 25, Ropa y accesorios)  **Product**(“B1234”, 5, “El Gran Gatsby”  “Una novela clásica ambientada en la década de 1920”., 15.99, 10, Libro) |
| Libros (enumeration) | **Product**((“B1235”, 5, “El principito”, “Una novela clásica de aventura e introspección”, 13.99, 20, Libros)  **Product**(“B1234”, 5, “El Gran Gatsby”  “Una novela clásica ambientada en la década de 1920”., 15.99, 10, Libro) |
| “Padre rico y padre pobre” | **null** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** verificar si la función de eliminación de producto en el inventario funciona correctamente, en caso de poner un código valido, otro para verificar si ya no puede eliminar porque ya fue eliminado antes y por ultimo si dicho producto no se encuentra en la base de datos. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| Store | deleteProduct() | setUp1() | B1235 | true |
| B1235 | false |
| B143 | false |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** verificar la función de actualizar la información de productos solicitando el código del producto, en caso de que el producto sea encontrado en la base de datos o no. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| Store | updateProduct() | setUp1() | **productCode:** “E5678”  **timesPurchased:** 25  **productName:** “Smartphone Galaxy S21+ stylus”  **productDescription: “**Un teléfono inteligente con pantalla AMOLED y cámara de alta resolución más un lápiz táctil”  **price:** “$699.99”  **stock:** 25  **productCategory:** Electrónica | true |
| **productCode:** “E5679”  **timesPurchased:** 20  **productName:** “Smartphone Galaxy S21”  **productDescription: “**Un teléfono inteligente con pantalla AMOLED y cámara de alta resolución.”  **price:** “$699.99”  **stock:** 30  **productCategory:** Electrónica | false |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** verificar la función de generar una orden, en el caso de que el producto se encuentre en la base de datos, de que no se encuentre y en caso de ingresar un método de pago no registrado previamente | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| Store | createOrder() | setUp()  setUp1() | **Nombre de usuario**: “user1”  **Contraseña**: “clave123”  **productID**: E5678  **quantity**: 5  **methodPayment**: CreditCard | true |
| **Nombre de usuario**: “user3”  **Contraseña**: “segura456”  **productID**: 67890  **quantity**: 3  **methodPayment**: PSE | False (don’t found) |
| **Nombre de usuario**: “user3”  **Contraseña**: “segura456”  **productID**: R9876  **quantity**: 2  **methodPayment**: CreditCard | False (do not have this payment method registered) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método falla cuando el arreglo contiene elementos que no son enteros. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| RadixSort | |  | | --- | | radixSort |  |  | | --- | |  | | Arreglo con elementos no enteros | Array(1, 2, "three", 4) | Error o Excepción |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método falla cuando el arreglo contiene valores null. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| RadixSort | |  | | --- | | radixSort |  |  | | --- | |  | | Arreglo con valores null | Array(170, 45, null, 90) | Error o Excepción |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método ordena correctamente un arreglo de enteros positivos. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| RadixSort | radixSort | Arreglo de enteros positivos | Array(170, 45, 75, 90, 802, 24, 2, 66) | Array(2, 24, 45, 66, 75, 90, 170, 802) |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método maneja correctamente un arreglo con elementos duplicados. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| RadixSort | radixSort | Arreglo con elementos duplicados | Array(170, 45, 75, 90, 802, 24, 75, 66) | Array(24, 45, 66, 75, 75, 90, 170, 802) |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método maneja correctamente un arreglo con un solo elemento. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| RadixSort | radixSort | Arreglo con un solo elemento | Array(42) | Array(42) |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método no altera un arreglo que ya está ordenado. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| RadixSort | radixSort | Arreglo ya ordenado | Array(1, 2, 3, 4, 5, 6, 7, 8, 9) | Array(1, 2, 3, 4, 5, 6, 7, 8, 9) |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método maneja correctamente un arreglo vacío. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| RadixSort | radixSort | Arreglo vacío | Array[Int]() | Array[Int]() |
|  |  |  |  |  |
|  |  |  |  |  |
| **Objetivo de la Prueba:** Verificar que el método maneja correctamente un arreglo con enteros positivos y negativos. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| RadixSort | radixSort | Arreglo con enteros positivos y negativos | Array(3, -2, 1, -5, 0) | Array(-5, -2, 0, 1, 3) |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método ordena correctamente un arreglo de enteros desordenados. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| HeapSort | heapSort | Arreglo desordenado | Array(4, 1, 3, 2, 16, 9, 10, 14, 8, 7) | Array(1, 2, 3, 4, 7, 8, 9, 10, 14, 16) |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método no altera un arreglo que ya está ordenado. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| HeapSort | heapSort | Arreglo ya ordenado | Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) | Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método ordena correctamente un arreglo que inicialmente está en orden descendente. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| HeapSort | heapSort | Arreglo en orden descendente | Array(10, 9, 8, 7, 6, 5, 4, 3, 2, 1) | Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método maneja correctamente un arreglo con elementos repetidos. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| HeapSort | heapSort | Arreglo con elementos repetidos | Array(4, 10, 3, 10, 2, 10, 1, 4) | Array(1, 2, 3, 4, 4, 10, 10, 10) |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método lanza una excepción al recibir un arreglo null. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| HeapSort | heapSort | Arreglo null | null | Error o Excepción |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método lanza una excepción al recibir un arreglo con tipos de datos mixtos. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| HeapSort | heapSort | Arreglo con tipos de datos mixtos | Array(5, "text", 3, 2.5) | Error o Excepción |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método ordena correctamente un arreglo de enteros desordenados. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| CountingSort | countingSort | Arreglo desordenado | Array(4, 1, 3, 2, 16, 9, 10, 14, 8, 7) | Array(1, 2, 3, 4, 7, 8, 9, 10, 14, 16) |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método no altera un arreglo que ya está ordenado. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| CountingSort | countingSort | Arreglo ya ordenado | Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) | Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método ordena correctamente un arreglo que inicialmente está en orden descendente. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| CountingSort | countingSort | Arreglo en orden descendente | Array(10, 9, 8, 7, 6, 5, 4, 3, 2, 1) | Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método maneja correctamente un arreglo con elementos repetidos. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| CountingSort | countingSort | Arreglo con elementos repetidos | Array(4, 10, 3, 10, 2, 10, 1, 4) | Array(1, 2, 3, 4, 4, 10, 10, 10) |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método lanza una excepción o falla al recibir un arreglo con enteros negativos (considerando que el algoritmo Counting Sort clásico no maneja números negativos). | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| CountingSort | countingSort | Arreglo con enteros negativos | Array(-1, -3, 4, 2) | Error o Excepción |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Verificar que el método lanza una excepción al recibir un arreglo con tipos de datos mixtos. | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| CountingSort | countingSort | Arreglo con tipos de datos mixtos | Array(5, "text", 3, 2.5) | Error o Excepción |
|  |  |  |  |  |
|  |  |  |  |  |

…

\* Una prueba se compone de un conjunto de casos de prueba.

\* Cada fila representa un ***caso de prueba*** diferente

\* En el objetivo de la prueba debe escribir una descripción sobre qué es lo que específicamente está probando del modelo del programa.

\* La clase es la clase del modelo que está siendo puesto a prueba.

\* El método es específicamente el método de la clase que está siendo puesto a prueba.

\* El escenario se refiere al nombre del escenario que usted definió. Todos los casos de prueba corresponden a escenarios.

\* Los valores de entrada son valores que entran al método puesto a prueba.

\* El resultado esperado es lo que se espera que suceda luego de ejecutar el método.