**Formato de escenarios y casos de uso**

**Configuración de los Escenarios**

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Clase** | **Escenario** |
| setupStage1 | QueueTest | Cola vacia |
| setupStage2 | QueueTest | Cola con 2 passengers object  Passenger p1( name:” Santiago” ,surname:” Hernandez” first Class:false, seat:6, miles:100, id: 1105362267)  object Passenger p2( name:” Sergio” ,surname:” florez” first Class: false, seat:15, miles:1000, id: 6543217890) |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **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:** Probar el correcto funcionamiento del método enqueue | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| Queue | enqueue | setupStage1 | Passenger p1( name:” Santiago” ,surname:” Hernandez” first Class:false, seat:6, miles:100, id: 1105362267) | Al imprimir el queue, nos debe imprimir “Santiago” |
| Queue | enqueue | setupStage2 | object Passenger p2( name:” Sergio” ,surname:” florez” first Class: false, seat:15, miles:1000, id: 6543217890)  Passenger p1( name:” Santiago” ,surname:” Hernandez” first Class:false, seat:6, miles:100, id: 1105362267) | Al imprimir el queue, nos debe imprimir  “Santiago  Sergio  Sergio  Santiago ” |
| Queue | enqueue | setupStage1 | object Passenger p2( name:” Sergio” ,surname:” florez” first Class: false, seat:15, miles:1000, id: 6543217890) | Al imprimir el queue, nos debe imprimir “Sergio” |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Probar el correcto funcionamiento de dequeue | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| Queue | dequeue | setupStage2 | Llamaremos el método una vez | Debería luego imprimir “Santiago” y debería tener un tamaño de 1 el queue |
| Queue | dequeue | setupStage2 | Llamaremos el método dos vez | Juntaríamos los 2 elementos y debería impirimir “Santiago Sergio” y debería tener un tamaño de 0 el queue |
| Queue | dequeue | setupStage1 | LLamariamos el método una vez | Debería soltar la excepción EmptyStackException(); |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Objetivo de la Prueba:** Probar el correcto funcionamiento de peek | | | | |
| **Clase** | **Método** | **Escenario** | **Valores de Entrada** | **Resultado esperado** |
| Queue | peek | setupStage2 | Llamaremos el método una vez | Debería luego imprimir “Santiago” y debería tener un tamaño de 2 el queue |
| Queue | peek | setupStage2 | Llamaremos el método dos vez | Juntaríamos los 2 elementos y debería impirimir “Santiago Sergio” y debería tener un tamaño de 2 el queue |
| Queue | peek | setupStage1 | LLamariamos el método una vez | Debería soltar la excepción EmptyStackException(); |

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

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

\* 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.