Diseño de pruebas unitarias

|  |  |
| --- | --- |
| Caso de Prueba | # 1 |
| Método | CargarClientes() |
| Objetivo | Probar que el método es capaz de realizar una carga correcta de los datos |
| Entrada | Un archivo excel “.cvs” con aproximadamente 4187 clientes |
| Salida | Lista <> de clientes no nula, con los diferentes clientes cargados de la base de datos. |

|  |  |
| --- | --- |
| Caso de Prueba | # 2 |
| Método | CargarArticulos() |
| Objetivo | Probar que el método es capaz de realizar una carga correcta de los datos |
| Entrada | Un archivo excel “.cvs” con aproximadamente 7390 artículos |
| Salida | Lista <> de Articulos no nula, con los diferentes artículos cargados de la base de datos. |

|  |  |
| --- | --- |
| Caso de Prueba | # 3 |
| Método | CargarVentas() |
| Objetivo | Probar que el método es capaz de realizar una carga correcta de los datos |
| Entrada | Un archivo excel “.cvs” con aproximadamente 149300 ventas |
| Salida | Lista <> de Ventas no nula, con las diferentes ventas cargadas de la base de datos. |

|  |  |
| --- | --- |
| Caso de Prueba | # 4 |
| Método | Combinacion() |
| Objetivo | Probar que el método es capaz de realizar correctamente las combinaciones en un arreglo |
| Entrada | Un arreglo de enteros {1,2,3} |
| Salida | Una List<List<int>> no nula, donde :  List1 { 1,2,3} , List2 { 1,3,2} , List3 {2,1,3} , List4 {2,3,1} , List5 {3,1,2} ,  List6 {3,2,1} |

,

|  |  |
| --- | --- |
| Caso de Prueba | # 5 |
| Método | repeticionesVentas() |
| Objetivo | Probar que el método es capaz de contar correctamente las repeticiones de las diferentes ventas que se encuentran en la base de datos |
| Entrada | Una lista<int> de la primera combinacion posible de la lista de combinaciones de los articulos mas frecuentes de la base de datos. |
| Salida | Int = 33 el cual representa la cantidad de repeticiones de esta combinación |