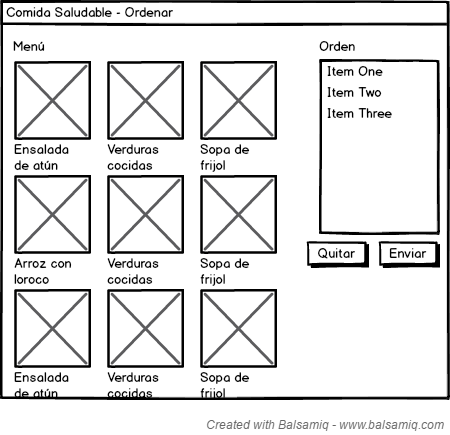
# http://vectorblog.org/wp-content/uploads/2013/06/Telehealth-cartoon-e1370345782337.jpgHoja de trabajo No. 3: *Listas*

Objetivos de la hoja de trabajo

* Identificar la definición de una *Lista*.
* Identificar diferencias entre una lista y una cola.
* Utilizar operaciones básicas de las listas.

Recomendación: trabajar en grupos de dos integrantes.

Recordemos que para la aplicación de *Pedidos y Cocina*, en la pantalla de Recepción de pedidos, se tenía la opción de agregar comidas a la orden y también una opción para quitar una comida de la orden.



1. ¿Qué mejoras cree que podría realizar en el programa de *Pedidos y Cocina* al utilizar listas?

* Al contar con las operaciones de acceso de las listas, podríamos darle al usuario mas flexibilidad, como por ejemplo borrar cualquier item que haya ordenado y no solo el último o el primero.

1. ¿Qué cree que el usuario prefiere: poder quitar solamente la última comida agregada o poder quitar cualquier comida, seleccionándola y luego pulsando el botón Quitar?

Definitivamente cualquier comida, incluso sería bueno poder eliminar varias al mismo con un solo click.

1. ¿Qué sería necesario para permitir al usuario eliminar cualquier comida de la orden? ¿Qué debería cambiar en la interfaz de usuario? ¿Qué debería cambiar en el código interno?

Primero sería necesario cambiar la estructura de datos que se este utilizando por una lista.

Sería necesario que el usuario pudiera seleccionar cualquier comida de las que ha ordenado. En el código interno utilizamos el indice de la comida seleccionada para eliminarla de la lista. Es importante o más bien crítico que ambas listas estén siempre sincronizadas, que sean paralelas, para que el programa funcione bien.

1. Considerando la lectura de la Definición de las listas (libro Data Structures and Program Design in C++, páginas 212 a la 216), ¿en qué se diferencia una lista de una cola?

Se diferencia en las operaciones que podemos realizar, en la manera en la que podemos acceder y modificar los datos.

Tome un tiempo para discutir sus respuestas con otro grupo.

Evalúe la clase List<T> definida en .NET Framework, referencia disponible en: <https://msdn.microsoft.com/en-us/library/6sh2ey19(v=vs.110).aspx>

1. ¿Hay una equivalencia de los operaciones definidas en el libro con métodos de la clase List<T>? Indique cuáles son equivalentes.

|  |  |
| --- | --- |
| Operaciones de una lista (tipo de dato abstracto) | Métodos de List<T> |
| Construct | List<T>(); |
| Empty |  |
| Full |  |
| Size | Count |
| Clear | Clear() |
| Insert | Insert(int, T) |
| Remove | RemoveAt(int); |
| Retrieve | Item[int] |
| Replace | Item[int] = int |
| Traverse | ForEach(Action<T>) |

1. ¿Qué algoritmo (en pseudocódigo) propone para la operación Replace?

list[index] = item

1. ¿Qué algoritmo (en pseudocódigo) propone para la operación Empty?

Return Count == 0

Tome un tiempo para discutir sus respuestas con otro grupo.

1. Suponga que se desea una operación *especial* que recibe de entrada dos listas y devuelve una lista formada por la “unión” de ambas listas. ¿Qué algoritmo (en pseudocódigo) propone para esta operación?

Añadir a la lista 1 todos los elementos de la lista 2.

1. Suponga que se desea una operación *especial* que recibe de entrada tres listas y como resultado, en la segunda lista tendrá los elementos pares de la primera lista. Y la tercera lista tendrá los elementos impares de la primera lista. ¿Qué algoritmo (en pseudocódigo) propone para esta operación?

For loop para iterar por todos los elementos, añadir los que sean pares a la lista 2, y los impares a la lista 3.

Tome un tiempo para discutir sus respuestas con otro grupo.