**Práctica No 4. Aplicaciones con listas**

Introducción

1. **Tipo de dato Abstracto (TDA o TAD)**

Un tipo de dato abstracto (TDA) o Tipo abstracto de datos (TAD) es un modelo matemático compuesto por una colección de operaciones definidas sobre un conjunto de datos para el modelo.

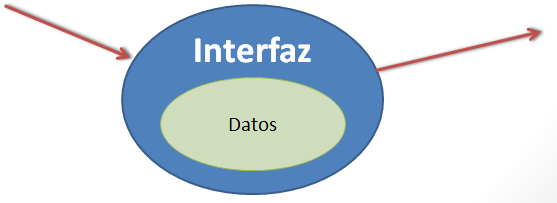
También se dice que es una entidad abstracta formada por un conjunto de datos organizados en cierta estructura y una colección de operaciones asociadas especificados formalmente.

Un TAD representa una abstracción de datos, es decir:

* Se destacan los detalles (normalmente pocos) de la especificación (¿el qué?).
* Se ocultan los detalles (casi siempre numerosos) de la implementación (¿el cómo?).

Cuando se usa en un programa de computación, un TAD es representado por su interfaz ***¿el qué?***, la cual sirve como cubierta a la correspondiente implementación ***¿el cómo?***

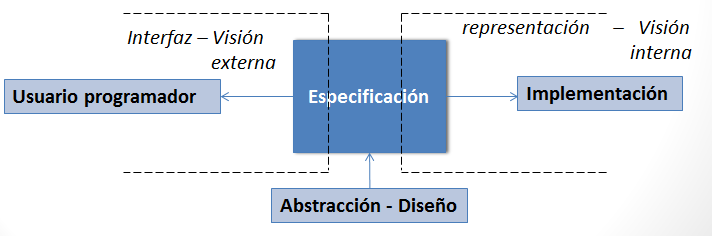
Un TAD utiliza el concepto de ocultación de información, ya que los usuarios tienen que preocuparse por la interfaz, pero no por la implementación. La solidez de un TAD radica en la idea de que la implementación está escondida al usuario. Solo la interfaz es pública.



E.g. el TAD puede ser implementado de diferentes formas, pero mientras se mantenga consistente con la interfaz, los programas que lo usan no se ven afectados.

* 1. **Especificación de un TAD**

En la especificación de un TAD se establecen sus propiedades, se define totalmente su comportamiento, pero no se dice nada sobre su implementación. Indica el tipo de entidades que modela, que operaciones se les pueden aplicar, como se usan y que hacen.



**Características de la especificación de un TAD**

* **Precisa**: Menciona únicamente aquello que es imprescindible.
* **General**: Se adapta a los diferentes contextos que se podrían llegar a manejar.
* **Legible**: Debe ser entendible para lograr una comunicación clara entre el especificador, el implementador y los usuarios del tipo.
* **No ambigua**: Que evite futuros problemas en la manera de interpretarse.

1. **Listas**

Una lista es un TAD lineal donde todos sus elementos son de un mismo tipo y estos se encuentran colocados uno detrás de otro. Cada elemento de la lista se conoce con el nombre de NODO de la lista.

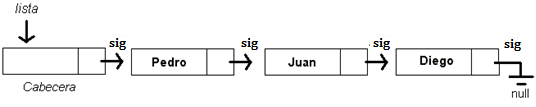
Una lista es también, una colección de 0 o más elementos, si la lista no tiene elementos, se dice que está vacía

Las listas son mucho más flexibles que los arreglos ya que permiten trabajo “dinámico” con un grupo de elementos. Es dinámica ya que se puede reservar memoria y liberarla en el momento que se requiera, es decir, una lista puede incrementar y decrementar su tamaño de forma arbitraria.

* 1. **Listas simplemente ligadas**

Una lista se constituye por una colección de elementos llamados nodos. El orden entre estos elementos se establece por punteros, es decir, por direcciones o referencias a otros nodos. En general un nodo consta de dos partes:

1. Un campo de INFORMACION que será del tipo de datos que se quieran almacenar en la lista
2. Un campo LIGA de tipo puntero, que se utiliza para establecer la liga o el enlace con otro nodo de la lista. Si fuera el último nodo de la lista, este campo tendría un valor NULO (NULL).



Una lista simplemente ligada tiene la peculiaridad de que el recorrido entre sus nodos se hace en un solo sentido, es decir, si se busca un elemento dentro de la lista el recorrido siempre será desde el inicio de la lista hasta que se encuentra el elemento buscado, en caso de que no se encuentre dicho elemento, no habrá forma de saberlo, sino hasta haber recorrido la lista completa.

Lo anterior se puede ver como una desventaja de la lista simplemente ligada, sin embargo, es la forma en que se comporta. Para quitar esta desventaja existen otras implementaciones de la lista, por ejemplo: la lista doblemente ligada o la lista simplemente ligada circular donde no hay un inicio en particular.

Desarrollo de la práctica

Una casa de empeño solicita a los alumnos de ESCOM que desarrollen un sistema para llevar el control de los clientes. Lo que requiere es tener una lista actualizada con los datos completos de cada cliente y una lista de los artículos empeñados por cada cliente. Es decir, cada cliente será un nodo de una lista y cada nodo tendrá enlazada otra lista para almacenar los artículos empeñados por el cliente, como se muestra en la siguiente figura:



Nota: Esta figura únicamente muestra la forma lógica de como estarían enlazados los nodos de las listas. Sin embargo, puede cambiar dependiendo del tipo de lista a utilizar.

El sistema deberá cumplir con los siguientes requerimientos:

**Menú 1**: Menú principal:

1. **Mostrar la lista de clientes**.- Esta opción mostrará todos los datos de cada uno de los clientes que el sistema tiene registrados.
2. **Dar de alta clientes**.- Esta opción permitirá, que dada la clave de un cliente, permita modificar los datos del cliente.
3. **Dar de baja clientes**.- Esta opción permitirá, que dada la clave de un cliente, eliminar al cliente.
4. **Modificar datos de cliente**.- Esta opción permitirá, que dada la clave de un cliente, modificar los datos del cliente.
5. **Ver bienes del cliente**.- Esta opción permitirá, que dada la clave de un cliente, enviar a un menú 2.
6. **Salir del programa**.

**Menú 2:** La opción **(e)** permitirá recibir la clave de un cliente y mostrar el siguiente menú:

1. **Ver bienes del cliente**.- Esta opción permitirá mostrar la lista de bienes que el cliente ha empeñado.
2. **Ingresar nuevo bien**.- Esta opción permitirá ingresar un nuevo bien a la lista del cliente.
3. **Modificar bien**.- Esta opción permitirá, que dada la clave de un bien, modificar los datos del bien.
4. **Eliminar bien**.- Esta opción permitirá eliminar un bien de la lista de bienes que el cliente tiene empeñados. El bien de un cliente se podrá eliminar de dos formas posibles. 1) El cliente ha pagado el monto por el cual dejo empeñado el bien y 2) Caduco el periodo que el cliente tenia para pagar el monto total de su bien, por lo tanto, el bien pasa a ser propiedad de la casa de empeño.
5. **Regresar al menú principal**.

Se deberán basar en las siguientes estructuras, sin embargo, podrán agregar los campos necesarios a cada una de ellas para que su aplicación funcione de manera correcta.

|  |  |
| --- | --- |
| /\*Esta estructura define los datos o las características de un cliente\*/  **typedef** struct**{**  char nombre**[**100**];**  /\*todos los demás campos necesarios que  definen los datos de un cliente\*/  **.** **.** **.**  **}**DatosCliente**;**  /\*Esta estructura define los nodos de la lista de clientes. Contiene un apuntador a un nodoBien que fungirá como cabecera de la lista de bienes\*/  **typedef** struct nodo**{**  DatosCliente datos**;**    nodoBien \*headBien; **;**/\*Apuntador que  hará referencia a la lista de bienes\*/  /\*los demás campos necesarios que definen  al nodo para el tipo de lista a utilizar\*/  **.** **.** **.**  **}**nodoCliente**;**  /\*Esta estructura define al TAD lista de clientes, es decir, dentro contiene un apuntador que va a hacer referencia al inicio de la lista\*/  **typedef** struct**{**  nodoL **\***headClientes**; ;**/\*Apuntador que  hará referencia a la lista de clientes\*/  /\*los campos necesarios que definen la  lista a utilizar\*/  **.** **.** **.**  **}**ListaClientes**;** | /\*Esta estructura define las características de un bien que el cliente puede empeñar\*/  **typedef** struct**{**  char nombre**[**100**];**  float costo**;**  /\*todos los demás campos necesarios que  definen las caracteristicas del bien\*/  **.** **.** **.**  **}**DatosBien**;**  /\*Esta estructura define los nodos de la lista de bienes\*/  **typedef** struct nodoBien**{**  DatosBien datos**;**  int ID**;**/\*Este campo define el lugar donde se va a insertar el bien dentro de la lista de bienes\*/  /\*los campos necesarios que definen al nodo  del tipo de lista a utilizar\*/  **.** **.** **.**  **}**nodoBien**;** |

# Consideraciones a tomar en cuenta:

1. Cada cliente o nodoCliente de la lista, deberá tener asociada una lista de bienes empeñados por el cliente.
2. Se deberán hacer pruebas con las diferentes listas que se tiene: 1) lista simplemente ligada, 2) lista doblemente ligada, 3) lista circular simplemente ligada, y si la alcanzamos a ver 4) lista circular doblemente ligada.

# Requerimientos de entrega de práctica:

1. La práctica será en equipos máximo de 3 personas.
2. Tomar en cuenta el formato de entrega de reporte de la práctica.
3. La fecha de revisión de la práctica será el lunes 17 de marzo de 2014 (queda pendiente el lugar de revisión para los grupos 1CV5 y 1CV7, venir preparados).
4. La fecha de entrega del reporte de la práctica será el miércoles 19 de marzo de 2014 hasta las 24:00. Reportes de prácticas fuera de horario no serán válidos.
5. Enviar un solo archivo comprimido que contenga el reporte de la práctica, los códigos fuente documentados. **NO ANEXAR CÓDIGOS EJECUTABLES o con extensión \*.exe**