UNLaM
Dto. Ingeniería e Investigaciones Tecnológicas

Tipos de Datos Abstractos (TDA) – LISTA.

Precondición para ver este documento: es necesario haber visto, trabajado y no queden dudas con los documentos anteriores, los correspondientes al TDA PILA y al TDA COLA.

Entendiendo el problema:

Con el TDA LISTA, se pueden modelar eventos, hechos o necesidades de la vida real y se pueden dar distintos ejemplos concretos. Los estudiantes de una comisión, una lista de palabras, los movimientos (extracciones y depósitos) bancarios, las cartas de un mazo con que se juega al veintiuno o a la canasta (varios mazos), o de una partida de póker (parte del mazo), etc., los clientes a los que hay que despacharles los pedidos, etcétera.

Un ejemplo trivial. Cada persona que llega a un evento llena en una ficha sus datos (nombre, apellido, entidad a la que pertenece, etc.). Durante el evento, se hace un sorteo y se extraen las fichas correspondientes a los premios de una bolsa (se extraen al azar). Terminado el evento, se debe informar a las entidades representadas, quienes han concurrido de cada entidad. El encargado de hacerlo ordena las fichas por entidad y luego por apellido y nombre.

Vamos a un ejemplo computacional: En un comercio mayorista se van cargado los pedidos con número de cliente, zona del cliente, código de producto, precio unitario, volumen y peso de cada producto y la cantidad pedida en un archivo secuencial. Al final del día, para armar los envíos se requiere armar un listado de distribución con el / los pedidos que se entregarán (por su volumen, peso y zona) en moto, mini flete, camión, etc. Este es un tema muy interesante porque en algunas zonas (como en microcentro), no se puede hacer, salvo en ciertos horarios, dependiendo del porte del transporte. Otro aspecto que sigue siendo muy interesante por el desafío que implica, es generar la hoja de ruta para cada transportista, en que intervienen múltiples factores.

Otro ejemplo: A partir del archivo de personal se requiere una lista de personas ordenada por DNI. Otra lista ordenada por mes y día del cumpleaños, apellido y nombre pero que

primero estén las mujeres y luego los hombres. Otra más que esté ordenada por antigüedad, apellido y nombre, pero sólo de los que tienen una antigüedad de más de cierta cantidad de años, meses y días, y de los que tienen una antigüedad menor a otra cantidad de años, meses y días (¡comienzan a correr los rumores de *reingeniería* en la organización!).

Todas estas operaciones sobre un TDA LISTA son posibles.

Con algo más de esfuerzo, valiéndose de dos pilas o de tres colas se puede ordenar información por la clave que se necesite, para luego tomar esa pila (o cola), como fuente de ingreso para otro ordenamiento (un buen artesano que no dispone de las mejores herramientas puede igualmente hacer su trabajo -a falta de plomada, piedra y "*piolín*", a falta de máquina de coser, aguja e hilo-). Pero con un TDA LISTA, estas operaciones son mucho más simples.

Recordemos que para los TDA PILA y TDA COLA sólo son válidas las siete operaciones que se indicaron (crear, vaciar, poner, sacar, determinar si está llena, saber si está vacía, ver el primero / tope).

En cambio, con el TDA LISTA, cualquier operación es posible, y si no la tiene en su biblioteca de primitivas de lista, podrá generar sus nuevas primitivas (u operaciones sobre la lista).

Primitivas elementales son las de crear, vaciar, poner, sacar, saber si está llena, saber si está vacía, poner al comienzo de la lista, ver el primero o sacarlo. Estas pocas primitivas ya las puede resolver con lo que sabe hasta ahora, pero igual las resolveremos a modo de "*revisión*".

Con lo dicho hasta ahora, ¿le basta para entender el problema? Como tiene solución, ¡ya verá que no es un problema!

Pasemos a ver la estrategia y solución a los diversos *problemas* o primitivas.

Observación, por una cuestión de estilo y buenas prácticas, las primitivas (las que se precien de serlo) no deberán involucrar otras primitivas. Deben involucrar la algoritmia requerida per se. En los casos en que haya código repetido en varias de ellas, y si se puede, podría resolverlo en un macro reemplazo. Esto hará más difícil el seguimiento y corrección ante alguna falla o error. En la fase de desarrollo, es conveniente trabajar con la norma de calidad de *cero defectos*, filosofía de trabajo muy conveniente para un desarrollador de software y ¡más aún para un estudiante!

NOTA: El modelo computacional del TDA LISTA, al igual que en los TDA PILA y TDA COLA, se puede implementar con asignación estática de memoria y con asignación dinámica de memoria. En la materia lo haremos tan sólo con asignación dinámica de memoria. Para una primera etapa, la implementación dinámica de memoria, la resolveremos mediante una lista simplemente enlazada, dejando para poco más adelante las listas dinámicas doblemente enlazadas.

OTRA: El modelo a tratar es el más sencillo. Trataremos con listas que harán referencia tan sólo al primer elemento. Hay modelos más sofisticados en que la lista hace referencia al primer elemento, al último insertado, al último eliminado o recuperado, y al último; con lo que estos modelos permiten operaciones más veloces en tiempo de ejecución además de ser conceptual y formalmente mucho más estrictas.

RECOMENDACIÓN: No hay modo de recordar o memorizar los algoritmos que vemos en la materia. Todos ellos, aún los que parezcan más triviales o que ya los haya resuelto o que formen parte de las explicaciones dadas (salvo que tenga mucha práctica), hay que volverlos a entender, determinar cuál de las posibles estrategias encarar, ayudarse de material concreto (cartas o esquemas) antes de comenzar a codificar (y también durante la misma). No deje de controlar que los algoritmos cumplan su cometido. No le deje al código (como decía un amigo) los "*flecos sueltos*". Y como dice otro, sepa qué va a hacer y cómo antes de ponerse a "*codear*".