TP U3 - Listas

Alumno: Nuñez Souto, Aaron Agustín

1. Una lista es una estructura dinámica de datos que contiene una colección de elementos homogéneos (del mismo tipo) de manera que se establece entre ellos un orden. Es decir, cada elemento, menos el primero, tiene un predecesor, y cada elemento, menos el último, tiene un sucesor.
2. Conceptos:
3. Lista Vinculada: es una estructura de datos compuesta de nodo. Cada nodo contiene alguna información, y un puntero a otro nodo de la lista. Si un nodo tiene un enlace solo a su sucesor en esta secuencia, la lista se denomina lista vinculada por si sola.
4. Lista Doblemente Vinculada: es un tipo de lista enlazada que permite moverse hacia adelante y hacia atrás. Cada nodo de una lista de este tipo tiene dos enlaces, además de los campos de datos. Un enlace, el derecho, se utiliza para navegar la lista hacia adelante. El otro enlace, el izquierdo, se utiliza para navegar la lista hacia atrás.
5. Lista Circular: en ésta, el primer y el último nodo están unidos juntos. Esto se puede hacer para listas enlazadas simples como para las dobles. Para recorrer una lista de este tipo, podemos comenzar por cualquier nodo y seguir la lista en cualquier dirección hasta que se regrese hasta el nodo original.
6. Lista Circular Doblemente Vinculada: cada nodo tiene 2 enlaces, similares a los de la lista doblemente enlazada, excepto que el enlace anterior del primer nodo apunta al último y el enlace siguiente del último nodo, apunta al primero.
7. Una de las ventajas de utilizar <LIST> de la STL, es que se implementan listas doblemente vinculada (con punteros en la cabeza y la cola). A su vez, generalmente funcionan mejor, que otros contenedores de secuencia   
   (díganse matriz, vector, entre otros), a la hora de insertar, extraer y mover elementos en cualquier posición dentro del contenedor para el que ya se ha obtenido un iterador y, por lo tanto, también en algoritmos que hacen un uso intensivo de estos.
8. Las operaciones básicas que implementan las listas son:
9. *splice*: Transfiere elementos de una lista a otra.
10. *remove*: Remueve elementos con valores específicos.
11. *remove\_if*: Remueve elementos de acuerdo a cierta condición.
12. *unique*: Remueve valores duplicados.
13. *merge*: Combina listas ordenadas.
14. *sort*: Ordena los elementos en el contenedor.
15. *reverse*: Invierte el orden de los elementos.
16. Formas de Manejo de Error a listas vacías:
17. if( !list.isEmpty() ){...} else { return 0; }
18. for ( ; head->next != tail; head = head->next);

b\*) variable temporal *tmp* que escanea la lista dentro del bucle for. La variable tmp se inicializa en el cabezal de la lista y, a continuación, en cada iteración del bucle, se avanza al siguiente nodo. Después de la segunda iteración y ejecutando la misma asignación, tmp se refiere al tercer nodo. Debido a que este nodo es también el siguiente al último nodo, se sale del bucle, después de lo cual se elimina el último nodo. Debido a que tail ahora apunta a un nodo no existe, se establece inmediatamente para apuntar al siguiente al último nodo al que actualmente apunta tmp.