

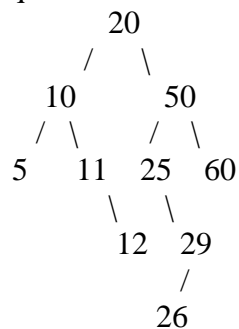
1.- Defina una estructura de datos que permita almacenar un diccionario de sinónimos Inglés-Español, Español-Inglés, satisfaciendo los siguientes requerimientos:

- (a) No duplique ningún string.
- (b) La estructura no puede ser estática.
- (c) Dado un string (palabra en Inglés o Español), encontrar eficientemente su equivalente en el otro idioma, en $O(\log_2 N)$.
- (d) Insertar una palabra junto a su equivalente en el otro idioma, debe hacerse con $O(\log_2 N)$.
- (e) Eliminar una palabra con $O(\log_2 N)$. Nota: Se debe eliminar su equivalente en el otro idioma, dentro del algoritmo de eliminación, de la manera más eficientemente posible.
- (f) Por cada palabra, debe existir una y sólo una palabra equivalente en el otro idioma.

Explique cómo su estructura de datos satisface los requerimientos.

2. Dado un árbol binario de enteros **CUALQUIERA**, realice un algoritmo que determine si es un árbol AVL

3. En el siguiente árbol binario de búsqueda:

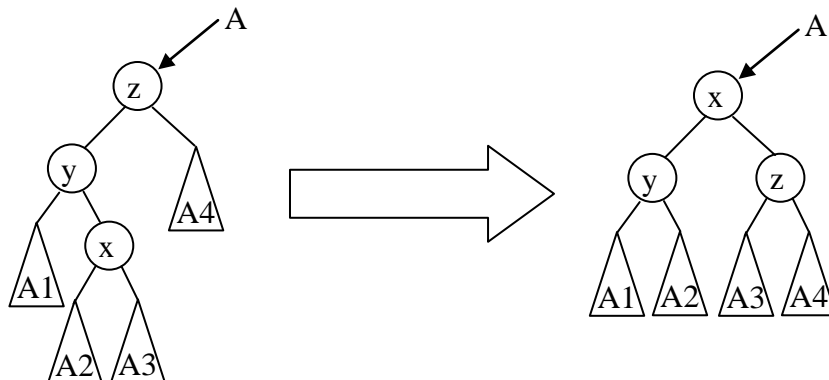


Usted debe realizar las siguientes operaciones, de forma NO secuencial (1 pt c/u):

- a) Insertar 30
- b) Eliminar 25
- c) Eliminar 10
- d) Eliminar 20
- e) Insertar 51

4. Para una clase MaxHeap, implemente una función “MaxHeap Mezclar (MaxHeap H)” que mezcle dos Montículos sin permitir elementos repetidos y sin destruir los heaps originales. Asuma que la clase Heap tiene implementadas las primitivas.

5. Dado un apuntador a la raíz de un subárbol, realice un algoritmo que implemente la “rotación doble”, tal y como se muestra en la figura:



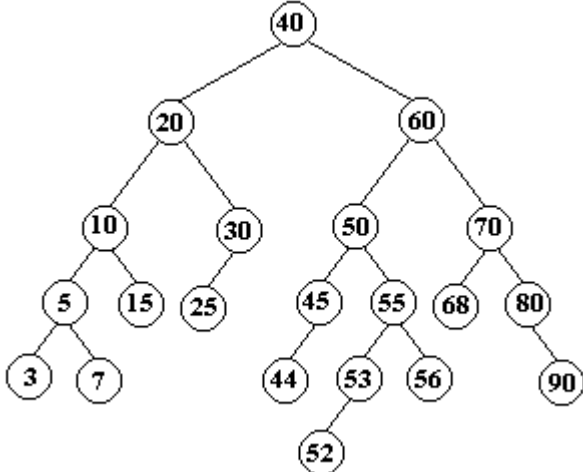
en donde A1, A2, A3 y A4 son árboles binarios que pueden ser vacíos o no vacíos. El procedimiento que realiza la rotación tiene el siguiente perfil:

Procedure RotarID(**Ref** Nodo ^A);

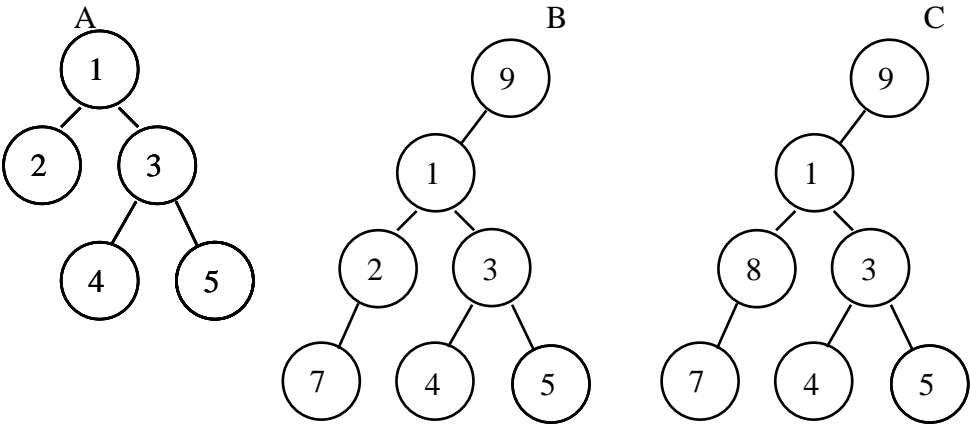
en donde Nodo está definido como:

Type Record Nodo
 Character x;
 Nodo ^izq;
 Nodo ^der;
End

6. Muestre gráficamente cómo queda el árbol AVL **A**, y qué tipo de rotación se efectúa al realizar sobre el árbol original cada una de las siguientes secuencias de operaciones

	<p>Secuencia a) A.Eliminar(30);</p> <p>Secuencia b) A.Insertar(8) A.Eliminar(68)</p> <p>Secuencia c) A.Eliminar(40)</p>
--	--

7. Dado dos árboles binarios A y B, realice un algoritmo que determine si un árbol está contenido dentro de otro, y en dado caso, indicar cuál.



En este ejemplo, A está contenido en B, pero no en C.

8. Diseñe un árbol balanceado, tal que al eliminar un determinado nodo se tengan que realizar dos rotaciones dobles. Realice las rotaciones sobre el árbol para justificar su respuesta.
9. Dado un árbol binario cualquiera (con la estructura de datos de un árbol binario cualquiera), realice un algoritmo que determine si el árbol está balanceado, pasando por cada nodo una sola vez.
10. Realice el algoritmo de recorrido en postorden de un árbol binario sin utilizar recursión. Ayuda: utilice una pila.