Programación y Estructuras de Datos (PED)

Examen sobre prácticas Junio 2014. GRADO.

Condiciones de entrega

- El examen se entrega a través del servidor de prácticas del DLSI **http://pracdlsi.dlsi.ua.es. Tras cada entrega, el servidor enviará al alumno un INFORME DE COMPILACIÓN, para que el alumno compruebe que lo que ha entregado cumple las especificaciones pedidas y que se ha podido generar el ejecutable correctamente. Este informe también se podrá consultar desde la página web de entrega de prácticas del DLSI (http://pracdlsi.dlsi.ua.es *e introducir el nombre de usuario y password*).**
- Se tiene que entregar un fichero comprimido **tgz** (**tar cvzf fichero.tgz** *) que contenga todos los ficheros de los cuadernillos (con la estructura de directorios especificada en el enunciado de la práctica: **dentro del .tgz solo deben aparecer los directorios lib, include y src**), junto con los métodos pedidos en el examen. El examen debe compilar con todos los ficheros entregados.
- El fichero nombres.txt tiene que contener el nombre del único autor del examen.
- El nombre de la función implementada por el alumno debe coincidir EXACTAMENTE con el prototipo propuesto en el enunciado.
- El alumno tiene que implementar su propio fichero de prueba (tad.cpp) para comprobar el código implementado (**este fichero no es necesario entregarlo**).
- El alumno puede añadir a la parte privada las variables y métodos que considere necesarios para la implementación.
- **SI SE ENTREGA ALGO QUE NO COMPILA SUPONDRÁ UN CERO EN EL EXAMEN. Solo se evaluará la salida del programa.**
- El enunciado se tiene que devolver una vez finalizado el examen.
- **ARCHIVOS A ENTREGAR (los 3 cuadernillos y la función *examen*)** → `include: tcomplejo.h, tvectorcom.h, tlistacom.h, tabbcom.h, tavlcom.h; lib: tcomplejo.cpp, tvectorcom.cpp, tlistacom.cpp, tabbcom.cpp, tavlcom.cpp`

Añadir la siguiente función a la parte pública de TABBCom:

bool examen(TListaCom &l)

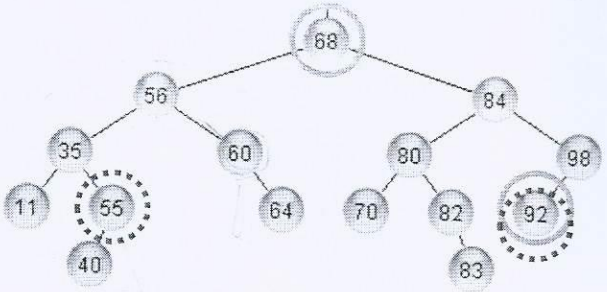
Dado un árbol **TABBCom** y una lista **TListaCom** pasada como parámetro, determinar si 2 de los nodos de la lista forman un CAMINO DESCENDENTE del ABB:

a) Si entre el MÍNIMO y el MÁXIMO nodo de la lista hay CAMINO DESCENDENTE en el ABB, se devuelve TRUE.
b) Si entre el MÍNIMO y el MÁXIMO nodo de la lista NO hay CAMINO DESCENDENTE en el ABB, se devuelve FALSE.

NOTAS :

1. **TListaCom** no está ordenada, así que el MÍNIMO y MÁXIMO nodo de la lista habrá que hallarlos mediante el criterio de ordenación propio de **TComplejo** (ya usado en la implementación de **TABBCom**).
2. Si el árbol o la lista de entrada son vacíos, se devuelve FALSE.

EJEMPLO (está hecho sólo con números enteros por simplificación):



TABBCom

Ejemplo 1 (SÍ existe el camino)

Lista entrada: { 68, 92, 84 }
Salida: TRUE

Ejemplo 2 (NO existe el camino)

Lista entrada: { 68, 92, 84, 55 }
Salida: FALSE