Muchach@s:    A continuación, una práctica general de **ARBOLES BINARIOS**:

Estudiantes: Manuel Calero y Brandon Mora.

**1) Qué es un árbol binario de búsqueda? Qué es un árbol binario normal ? Use los conceptos Raíz, Nodo Interno, Hoja, Hijos, SubArbolIzquierdo, SubarbolDerecho…**

* **Árbol binario de búsqueda: Árbol cuyos elementos está puestos en orden.**
* **Árbol binario normal: Árbol cuyos elementos está puestos en cualquier orden.**
* **Raiz: Es el elemento del cual parten las hojas y se van ordenando de manera que los valores que sean menores a esta se ubiquen a su izquierda y los mayores a su derecha, formando así a sus hijos. Una vez que uno de sus hijos tenga una hoja, este hijo se convertirá también en raíz.**
* **Nodo Interno:**
* **Hojas: Son los hijos de las diferentes raíces, estos hijos NO poseen ningún hijo.**
* **Hijos: Son los valores que se le asignan a las diferentes raíces, ya sean izquierdas o derechas, pero siempre manteniendo un orden.**
* **SubArbolIzquierdo: Son todo el conjunto de valores ubicados a la izquierda de la raíz principal, estos SubArboles también tienen raíces.**
* **SubarbolDerecho: Son todo el conjunto de valores ubicados a la derecha de la raíz principal, estos SubArboles también tienen raíces.**

**2) Por qué se puede decir que un árbol es más eficiente que una lista enlazada:**

* **Porque, pueden buscar los elementos con mayor rapidez ya que no es una búsqueda lineal, por lo tanto no necesita recorrer todos los elementos para encontrar uno solo.**

**Buenas, profe, queremos hacerle una aclaración respecto a la documentación del código de arboles binarios, le queremos que creamos funciones idénticas a las que usted nos dio, pero le hicimos cambios en la parte del print para que pudiese imprimir caracteres, por lo que se puede encontrar funciones como: EnOrdenIRDCh, que es igual a EnOrdenIRD. En el caso de las funciones que hacen tienen Ch no las comentamos porque hacen el mismo procedimiento que su función original cambiando únicamente en el print.**

La otra parte se encuentra dentro del Código.

**3)Con el adjunto arbolbb.cpp enviado:**

* ***Ejecutar el código fuente del demo de árboles binarios de búsqueda. Ojo recuerden observar con sumo cuidado los "returns"explícitos e implícitus de la recursión.  Ejecutar paso a paso en el laboratorio y documentar***

**3) Inserte en un arbol Binario de Búsqueda** **(3,7,5,1,6,8,4,2,9). Muestre el proceso paso a paso y** **finalmente del arbol generado indique con cuál** **pareja de recorridos coincide**

**a) Pre(321754689), Pos(124659873)**

**b) Pre(312754689), Pos(214659873)**

**c) Pre(321754689), En (123456789)**

**d) En (123456789), Pos(124659873)**

**4) Dados estos 2 recorridos genere el UNICO arbol que cumple:**

**a) PREORDEN : 9.5.1.4.3.15.11.14.13**

**b) POSORDEN 3.4.1.5.13.14.11.15.9**

5) Desarrollo :::: Implemente en C++ lo siguiente:

1. ***Una función para "contar" los elementos que tiene un árbol binario(Puede usar recorridos)***
2. ***Una función para comparar 2 árboles: devuelve true si son idénticos o false en caso contrario (Puede usar recorridos)***
3. ***Una función que "clone" un árbol en otro (Puede usar recorridos)***
4. ***Escriba una función que dado un árbol binario devuelva verdadero si el árbol es completo y falso en otro caso. Un árbol binario es completo si todos sus nodos tienen dos descendientes, excepto las hojas.***
5. ***Escriba una función que visualice los nodos que son hojas en un árbol binario.***
6. ***EXTRA … El método borrar un elemento de un árbol binario(opcional… pueden hacerlo en pseudocódigo)***

6)    Inserte la siguientes secuencias de Elementos en los árboles Binario de Búsqueda en la computadora(usando el programa arbolbb.cpp) y en papel

a.    ABB: 65 50 23 70 82 68 39

b.    ABB 43 58 75 86 65 70 67 73 93 69 25 66 68 47 62 10 60

c.    ABB: 10 27 29 17 25 21 15 31 13 51 20 24 48 19 60 35 66

d.    ABB: 10 68 80 27 53 43 21 77 58 63 15 37 41 72 39 95 70

7)    Regenere el árbol correspondiente a partir de estos recorridos:

a.    BinarioNormal(No ordenado)

 i.    Preorden : A B D E C F G

 ii.    InOrden: D B E A F C G

iii.    PostOrden : D E B F G C A

b.    Binario de Búsqueda:

 i.    Preorden : 120 87 43 22 65 56 99 93 140 130 135

ii.    InOrden: 22 43 56 65 87 93 99 120 130 135 140

iii.    PostOrden : 22 56 65 43 93 99 87 135 130 140 120

c.    Binario de Búsqueda:

i.    Preorden : 95 80 72 60 82 81 84 100 110 105

ii.    InOrden:  60 72 80 81 82 84 95 100 105 110

iii.    PostOrden : 60 72 81 84 82 80 105 110 100 95

Prof. Víctor Garro

