BÀI TẬP CẤU TRÚC DỮ LIỆU

Bài 1: Vẽ cây AVL tạo thành bằng cách thêm lần lượt các khóa sau (vẽ cây trước và sau mỗi lần thực hiện phép quay)

a. 10, 30, 35, 32, 20, 8
b. 12, 34, 53, 76, 15, 21, 18, 45, 16, 55, 11
c. 1, 3, 5, 7, 9, 12, 15, 17, 21, 23, 25, 27

Lưu ý:

- Chiều cao cây rỗng bằng -1: height(null)=-1
- Chiều cao nút lá bằng : height(leaves)=0
- Chiều cao cây : height(T)= max (height(T->Left), height(T->Right)+1

Bài 2: Xây dựng thư viện cây AVL

a. Khai báo cấu trúc AVLTree. Trong đó mỗi nút bao gồm: khóa (key), trạng thái cân bằng của nút (bal), chiều cao nút (height), con trái left, con phải (right).
b. Định nghĩa các trạng thái cân bằng: BALANCE=0 (cân bằng), LEFT=1 (lệch trái); RIGHT=2 (lệch phải);
c. Vẽ cây kết quả của các câu a, b, và C ở bài 1 với giá trị khóa và trạng thái cân bằng (0, 1 hay 2) và chiều cao của mỗi nút trên cây.
d. Viết hàm tạo nút mới có khóa x: createAVLNode(ElementType x){…}
e. Viết hàm trả về độ cao của nút: int height(AVLNode node){…}
f. Viết hàm quay trái: rotateLeft(AVLNode *pNode){…}
g. Viết hàm quay phải: rotateRight(AVLNode *pNode){…}
h. Viết hàm quay trái-phải (L-R rotate): rotateLeftRight(AVLNode *pNode){…}
i. Viết hàm quay phải-trái (R-L rotate): rotateRightLeft(AVLNode *pNode){…}
j. Viết hàm thêm khóa x vào cây AVL: insertNode(ElemenType x, AVLTree *root){…}
k. Viết các hàm duyệt preOrder(AVLTree root), inOrder(AVLTree root), PosOrder(AVLTree root) và levelOrder(AVLTree root) liệt kê khóa và trạng thái cân bằng của mỗi nút.
l. Viết hàm main() dựng các cây AVL ở câu a, b và C ở Bài 1 và thực hiện các phép duyệt trên cây.