KIM
KING
LION
KIND
JADE

(1) Several Primitive Operations on Shin's Tree

1. Search a key in the Shin tree.
2. Print inserted keys in the Shin tree in sorted order.
3. Insert a key into the tree.
4. Traverse the Shin tree in 4 different ways:
    a. preorder
    b. suorder
    c. inorder
    d. postorder
5. Convert Shin tree to its corresponding reverse one and vice versa.
6. Delete a key from the tree.

(2) Pseudo-code for Shin_Sort_The_Key routine

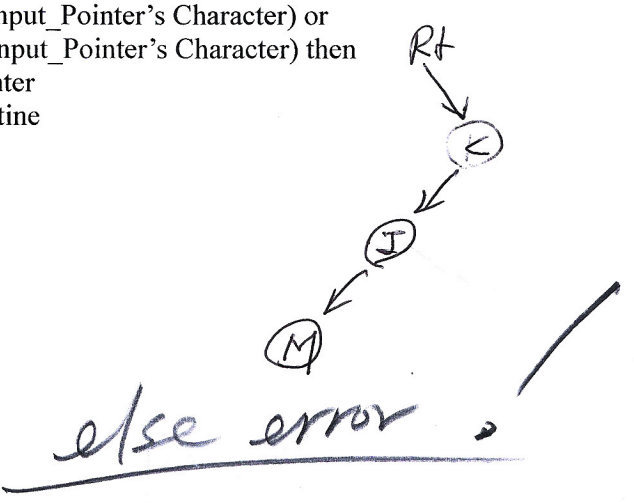
{The Routine that calls Shin Sort recursive routine. It creates Shin's tree initially; two pointer parameters: one for Shin tree and one for input key. It calls shin sort recursive routine to insert key into created Shin tree.}

Read_and Store_Input_String_and_Get_the_Pointer {input nodes linked through left son pointers; if they are integer number, sufficient zero characters are filled in for each integer number beforehand.}

If Root_Pointer = nil then Root_Pointer <== Input_Pointer

else If Root_Pointer's Character > Input_Pointer's Character then
    Input_Pointer's Right_Child <== Root_Pointer
    Root_Pointer <== Input_Pointer to the first character in key string

else If (Root_Pointer's Character < Input_Pointer's Character) or
    (Root_Pointer's Character = Input_Pointer's Character) then
    Tree_Pointer <== Root_Pointer
    Call Shin_Sort recursive routine



KIM
KING
LION
KIND
JADE
KIN
KENT
KILL
QUEEN

(3) Pseudo-code for Shin_Sort recursive routine

{This recursive Shin_Sort module will insert input key into right position in created Shin tree by comparing pointed input character with pointed tree node's character. Based on the comparison: equal, less, and greater, it performs appropriate action.}

Recursive Shin_Sort Routine (Root_Pointer, Input_Pointer);

If Tree_Pointer = nil then Header_Pointer' left or right son gets Current Input Node's pointer based on preset value of left/right flag (Header_Left_Right_Flag)

Else if Pointed_Input_Character = Pointed_Tree_Character then
    Header_Pointer <== Tree_Pointer
    Header_Left_Right_Flag <== left
    Tree_Pointer <== Tree_Pointer's left son
    If (Tree_Pointer = nil) or no_more_input_characters_left then
        Header-Pointed node's counter gets incremented by one
        If Tree_Pointer = nil then
            Input_Pointer goes to next character in the input key
            Header_Pointer's left child <== Input_Pointer
        Else
            Input_Pointer goes to next character in the input key
            Call itself, the Shin_Sort recursive routine to move in the tree

revised on 11/19/2009

Else if Input_Character < Pointed_Tree_Character then insertion of the input node by:
    Based on Header_Left_Right_Flag
        If left then Header_Pointer's left son gets Input_Pointer
        If right then Header_Pointer's right son gets Input_Pointer
    Input_Pointer's Right Child gets Tree_Pointer to complete the insertion of the node to tree

Else if Input_Character > Pointed_Tree_Character then
    Header_Pointer <== Tree_Pointer
    Set Header_Left_Right_Flag to right
    Tree_Pointer <== Tree_Pointer's right child
    Call itself, the Shin_Sort recursive routine to move to the right place in the tree