1. AVL Tree operations using linked list

Description: Implements insertion, deletion (optional simplified), and inorder traversal for an AVL tree using dynamic memory (nodes with left/right pointers).

```c
// avl_tree.c
#include <stdio.h>
#include <stdlib.h>

typedef struct Node {
    int key;
    struct Node *left, *right;
    int height;
} Node;

int max(int a,int b){return (a>b)?a:b;}
int height(Node *n){ return n? n->height:0; }
Node* newNode(int key){
    Node* node = (Node*)malloc(sizeof(Node));
    node->key = key; node->left = node->right = NULL; node->height = 1;
    return node;
}

Node *rightRotate(Node *y){
    Node *x = y->left; Node *T2 = x->right;
    x->right = y; y->left = T2;
    y->height = max(height(y->left), height(y->right))+1;
    x->height = max(height(x->left), height(x->right))+1;
    return x;
}
Node *leftRotate(Node *x){
    Node *y = x->right; Node *T2 = y->left;
    y->left = x; x->right = T2;
    x->height = max(height(x->left), height(x->right))+1;
    y->height = max(height(y->left), height(y->right))+1;
    return y;
}
int getBalance(Node *n){ return n? height(n->left)-height(n->right):0; }

Node* insert(Node* node, int key){
    if(!node) return newNode(key);
    if(key < node->key) node->left = insert(node->left,key);
    else if(key > node->key) node->right = insert(node->right,key);
    else return node;
    node->height = 1 + max(height(node->left), height(node->right));
    int balance = getBalance(node);
    if(balance>1 && key < node->left->key) return rightRotate(node);
    if(balance<-1 && key > node->right->key) return leftRotate(node);
```

```
    if(balance>1 && key > node->left->key){ node->left =
leftRotate(node->left); return rightRotate(node);}
    if(balance<-1 && key < node->right->key){ node->right =
rightRotate(node->right); return leftRotate(node);}
    return node;
}

void inorder(Node* root){
    if(root){ inorder(root->left); printf("%d ", root->key);
inorder(root->right);} }

int main(){
    Node *root = NULL;
    int n, x;
    printf("Enter number of elements to insert: "); scanf("%d", &n);
    for(int i=0;i<n;i++){ printf("Enter key: "); scanf("%d", &x); root =
insert(root,x); }
    printf("Inorder traversal of AVL tree: "); inorder(root); printf("\n");
    return 0;
}
```
