#include<stdio.h>

#include<stdlib.h>

struct node

{

    int key;

    struct node \*left, \*right;

};

// A utility function to create a new BST node

struct node \*newNode(int item)

{

    struct node \*temp =  (struct node \*)malloc(sizeof(struct node));

    temp->key = item;

    temp->left = temp->right = NULL;

    return temp;

}

// A utility function to do inorder traversal of BST

void inorder(struct node \*root)

{

    if (root != NULL)

    {

        inorder(root->left);

        printf("%d \n", root->key);

        inorder(root->right);

    }

}

/\* A utility function to insert a new node with given key in BST \*/

struct node\* insert(struct node\* node, int key)

{

    /\* If the tree is empty, return a new node \*/

    if (node == NULL) return newNode(key);

    /\* Otherwise, recur down the tree \*/

    if (key < node->key)

        node->left  = insert(node->left, key);

    else if (key > node->key)

        node->right = insert(node->right, key);

    /\* return the (unchanged) node pointer \*/

    return node;

}

// Driver Program to test above functions

int main()

{

    /\* Let us create following BST

              50

           /     \

          30      70

         /  \    /  \

       20   40  60   80 \*/

    struct node \*root = NULL;

    root = insert(root, 50);

    insert(root, 30);

    insert(root, 20);

    insert(root, 40);

    insert(root, 70);

    insert(root, 60);

    insert(root, 80);

    // print inoder traversal of the BST

    inorder(root);

    return 0;

}