//C program to Implement a Binary Search Tree using Linked List and perform traversal operations.

#include <stdio.h>

#include <stdlib.h>

struct node

{

char data; //to store data

struct node \*right\_child; // right child

struct node \*left\_child; // left child

};

struct node\* new\_node(char data)

{

struct node \*temp; // node

temp = malloc(sizeof(struct node)); // allocating space to temp

temp->data = data; // assinging data to temp

temp->left\_child = NULL; // making children NULL

temp->right\_child = NULL;

return(temp); // returning the newly made node

}

void preorder(struct node \*root)

{

if(root!=NULL) // checking if the root is not null

{

printf(" %c ", root->data); // printing data at root

preorder(root->left\_child); // visiting left child

preorder(root->right\_child); // visiting right child

}

}

void postorder(struct node \*root)

{

if(root!=NULL) // checking if the root is not null

{

postorder(root->left\_child); // visiting left child

postorder(root->right\_child); // visiting right child

printf(" %c ", root->data); // printing data at root

}

}

void inorder(struct node \*root)

{

if(root!=NULL) // checking if the root is not null

{

inorder(root->left\_child); // visiting left child

printf(" %c ", root->data); // printing data at root

inorder(root->right\_child);// visiting right child

}

}

int main()

{

struct node \*root; //new structure

root = new\_node('D'); // making a new node

root->left\_child = new\_node('A'); //left child of root

root->right\_child = new\_node('F'); //right child of root

root->left\_child->left\_child = new\_node('E'); // new node

root->left\_child->right\_child = new\_node('B'); // new node

printf("Preorder:\n");

preorder(root);

printf("\n");

printf("Postorder:\n");

postorder(root);

printf("\n");

printf("Inorder:\n");

inorder(root);

printf("\n");

return 0;

}