```c
/**
 * 연결자료구조로 구현한 트리
 *
 */
#include <stdio.h>
#include <stdlib.h>
#include <memory.h>

typedef struct treeNode
{ // 연결 자료구조로 구성하기 위해 트리의 노드 정의
    char data;
    struct treeNode *left;  // 왼쪽 서브 트리에 대한 링크 필드
    struct treeNode *right; // 오른쪽 서브 트리에 대한 링크 필드
} treeNode;

// data를 루트 노드로 하여 왼쪽 서브 트리와 오른쪽 서브 트리를 연결하는 연산
treeNode *createRootNode(char data, treeNode *leftNode, treeNode *rightNode)
{
    treeNode *root = (treeNode *)malloc(sizeof(treeNode));
    root->data = data;
    root->left = leftNode;
    root->right = rightNode;
    return root;
}

// 이진 트리에 대한 전위 순회 연산
void preorder(treeNode *root)
{
    if (root)
    {
        printf("%c", root->data);
        preorder(root->left);
        preorder(root->right);
    }
}

// 이진 트리에 대한 중위 순회 연산
void inorder(treeNode *root)
{
    if (root)
    {
        inorder(root->left);
        printf("%c", root->data);
        inorder(root->right);
    }
}

// 이진 트리에 대한 후위 순회 연산
void postorder(treeNode *root)
{
```

```
    if (root)
    {
        postorder(root->left);
        postorder(root->right);
        printf("%c", root->data);
    }
}

void main()
{
    // (A*B-C/D) 수식 이진 트리 만들기
    treeNode *n7 = createRootNode('D', NULL, NULL);
    treeNode *n6 = createRootNode('C', NULL, NULL);
    treeNode *n5 = createRootNode('B', NULL, NULL);
    treeNode *n4 = createRootNode('A', NULL, NULL);
    treeNode *n3 = createRootNode('/', n6, n7);
    treeNode *n2 = createRootNode('*', n4, n5);
    treeNode *n1 = createRootNode('-', n2, n3);

    printf("\n preorder : ");
    preorder(n1);

    printf("\n inorder : ");
    inorder(n1);

    printf("\n postorder : ");
    postorder(n1);

    getchar();
}
```