/\*

240506 트리

\*/

1. **순회 프로그램 (순환)**

#include <stdio.h>

#include <stdlib.h>

#include <memory.h>

#define \_CRT\_SECURE\_NO\_WARNINGS

#pragma warning(disable:4996)

typedef struct TreeNode {

int data;

struct TreeNode\* left, \* right;

} TreeNode;

// 15

// 4 20

// 1 16 25

TreeNode n1 = { 1, NULL, NULL };

TreeNode n2 = { 4, &n1, NULL };

TreeNode n3 = { 16, NULL, NULL };

TreeNode n4 = { 25, NULL, NULL };

TreeNode n5 = { 20, &n3, &n4 };

TreeNode n6 = { 15, &n2, &n5 };

TreeNode\* root = &n6;

// 중위 순회

void inorder(TreeNode\* root) {

if (root != NULL) {

inorder(root->left);// 왼쪽서브트리 순회

printf("[%d] ", root->data); // 노드 방문

inorder(root->right);// 오른쪽서브트리 순회

}

}

// 전위 순회

void preorder(TreeNode\* root) {

if (root != NULL) {

printf("[%d] ", root->data); // 노드 방문

preorder(root->left);// 왼쪽서브트리 순회

preorder(root->right);// 오른쪽서브트리 순회

}

}

// 후위 순회

void postorder(TreeNode\* root) {

if (root != NULL) {

postorder(root->left);// 왼쪽서브트리 순회

postorder(root->right);// 오른쪽서브트리순회

printf("[%d] ", root->data); // 노드 방문

}

}

int main(void)

{

printf("중위 순회=");

inorder(root);

printf("\n");

printf("전위 순회=");

preorder(root);

printf("\n");

printf("후위 순회=");

postorder(root);

printf("\n");

return 0;

}

순회 프로그램[반복]

#include <stdio.h>

#include <stdlib.h>

#include <memory.h>

#define \_CRT\_SECURE\_NO\_WARNINGS

#pragma warning(disable:4996)

typedef struct TreeNode {

int data;

struct TreeNode\* left, \* right;

} TreeNode;

#define SIZE 100

int top = -1;

TreeNode\* stack[SIZE];

void push(TreeNode\* p)

{

if (top < SIZE - 1)

stack[++top] = p;

}

TreeNode\* pop()

{

TreeNode\* p = NULL;

if (top >= 0)

p = stack[top--];

return p;

}

void inorder\_iter(TreeNode\* root)

{

while (1) {

for (; root; root = root->left)

push(root);

root = pop();

if (!root) break;

printf("[%d] ", root->data);

root = root->right;

}

}

// 15

// 4 20

// 1 16 25

TreeNode n1 = { 1, NULL, NULL };

TreeNode n2 = { 4, &n1, NULL };

TreeNode n3 = { 16, NULL, NULL };

TreeNode n4 = { 25, NULL, NULL };

TreeNode n5 = { 20, &n3, &n4 };

TreeNode n6 = { 15, &n2, &n5 };

TreeNode\* root = &n6;

int main(void)

{

printf("중위 순회=");

inorder\_iter(root);

printf("\n");

return 0;

}

[수식트리 알고리즘]

#include <stdio.h>

#include <stdlib.h>

#define \_CRT\_SECURE\_NO\_WARNINGS

#pragma warning(disable:4996)

typedef struct TreeNode {

int data;

struct TreeNode\* left, \* right;

} TreeNode;

// +

// \* +

// 1 4 16 25

TreeNode n1 = { 1, NULL, NULL };

TreeNode n2 = { 4, NULL, NULL };

TreeNode n3 = { '\*', &n1, &n2 };

TreeNode n4 = { 16, NULL, NULL };

TreeNode n5 = { 25, NULL, NULL };

TreeNode n6 = { '+', &n4, &n5 };

TreeNode n7 = { '+', &n3, &n6 };

TreeNode\* exp = &n7;

// 수식 계산 함수

int evaluate(TreeNode\* root)

{

if (root == NULL)

return 0;

if (root->left == NULL && root->right == NULL)

return root->data;

else {

int op1 = evaluate(root->left);

int op2 = evaluate(root->right);

printf("%d %c %d을 계산합니다.\n", op1, root->data, op2);

switch (root->data) {

case '+':

return op1 + op2;

case '-':

return op1 - op2;

case '\*':

return op1 \* op2;

case '/':

return op1 / op2;

}

}

return 0;

}

int main(void)

{

printf("수식의 값은 %d입니다. \n", evaluate(exp));

return 0;

}

[디렉토리 용량 계산]

#include <stdio.h>

#include <stdlib.h>

#define \_CRT\_SECURE\_NO\_WARNINGS

#pragma warning(disable:4996)

typedef struct TreeNode {

int data;

struct TreeNode\* left, \* right;

} TreeNode;

int calc\_dir\_size(TreeNode\* root)

{

int left\_size, right\_size;

if (root == NULL) return 0;

left\_size = calc\_dir\_size(root->left);

right\_size = calc\_dir\_size(root->right);

return (root->data + left\_size + right\_size);

}

int main(void)

{

TreeNode n4 = { 500, NULL, NULL };

TreeNode n5 = { 200, NULL, NULL };

TreeNode n3 = { 100, &n4, &n5 };

TreeNode n2 = { 50, NULL, NULL };

TreeNode n1 = { 0, &n2, &n3 };

printf("디렉토리의 크기=%d\n", calc\_dir\_size(&n1));

}