

자료구조

**담당교수님 | 김희완**

**학과 | 컴퓨터메카트로닉스공학**

**주전공 | 컴퓨터공학전공**

**학년 | 2학년**

**학번 | 2019100994**

**이름 | 이정민**

**제출일 | 2020/05/24**

이진트리 구현 프로그래밍과제

-소스코드 (C)

#include <stdio.h>

#include <stdlib.h>

#include <memory.h>

typedef struct TreeNode {

struct TreeNode\* leftchild;

char data;

struct TreeNode\* rightchild;

}TreeNode;

TreeNode\* malloc\_Node(char data\_malloc, TreeNode\* leftchild, TreeNode\* rightchild) {

TreeNode\* node = (TreeNode\*)malloc(sizeof(TreeNode));

node->data = data\_malloc;

node->leftchild = leftchild;

node->rightchild = rightchild;

return node;

}

void Preorder(TreeNode\* root); void Inorder(TreeNode\* root); void Postorder(TreeNode\* root); void Norder\_address(TreeNode\* root);

int main() {

int choice;

printf("[이진트리 구현 프로그램]\n\n");

TreeNode\* n31 = malloc\_Node('P', NULL, NULL);

TreeNode\* n30 = malloc\_Node('O', NULL, NULL);

TreeNode\* n29 = malloc\_Node('N', NULL, NULL);

TreeNode\* n28 = malloc\_Node('M', NULL, NULL);

TreeNode\* n27 = malloc\_Node('L', NULL, NULL);

TreeNode\* n26 = malloc\_Node('K', NULL, NULL);

TreeNode\* n25 = malloc\_Node('J', NULL, NULL);

TreeNode\* n24 = malloc\_Node('I', NULL, NULL);

TreeNode\* n23 = malloc\_Node('H', NULL, NULL);

TreeNode\* n22 = malloc\_Node('G', NULL, NULL);

TreeNode\* n21 = malloc\_Node('F', NULL, NULL);

TreeNode\* n20 = malloc\_Node('E', NULL, NULL);

TreeNode \* n19 = malloc\_Node('D', NULL, NULL);

TreeNode \* n18 = malloc\_Node('C', NULL, NULL);

TreeNode \* n17 = malloc\_Node('B', NULL, NULL);

TreeNode \* n16 = malloc\_Node('A', NULL, NULL);

TreeNode \* n15 = malloc\_Node('-', n30, n31);

TreeNode\* n14 = malloc\_Node('+', n28, n29);

TreeNode\* n13 = malloc\_Node('-', n26, n27);

TreeNode\* n12 = malloc\_Node('+', n24, n25);

TreeNode\* n11 = malloc\_Node('-', n22, n23);

TreeNode\* n10 = malloc\_Node('+', n20, n21);

TreeNode\* n9 = malloc\_Node('-', n18, n19);

TreeNode\* n8 = malloc\_Node('+', n16, n17);

TreeNode\* n7 = malloc\_Node('/', n14, n15);

TreeNode\* n6 = malloc\_Node('\*', n12, n13);

TreeNode\* n5 = malloc\_Node('/', n10, n11);

TreeNode\* n4 = malloc\_Node('\*', n8, n9);

TreeNode\* n3 = malloc\_Node('-', n6, n7);

TreeNode\* n2 = malloc\_Node('+', n4, n5);

TreeNode\* n1 = malloc\_Node('\*', n2, n3);

**/\* 이진트리의 논리적구조**

**\***

**+ -**

**\* / \* /**

**+ - + - + - + -**

**A B C D E F G H I J K L M N O P**

**\*/**

printf(" -전위순회 >> 1 \n -중위순회 >> 2\n -후위순회 >> 3\n -모두출력 >> 4\n -주소값출력 >> 5\n 을 입력하세요 : ");

scanf\_s("%d", &choice);

switch (choice)

{

case 1:

printf("\n전위순회 출력 : "); //[\*][+][\*][+][A][B][-][C][D][/][+][E][F][-][G][H][-][\*][+][I][J][-][K][L][/][+][M][N][-][O][P]

Preorder(n1);

printf("출력완료");

break;

case 2:

printf("\n중위순회 출력 : "); //[A][+][B][\*][C][-][D][+][E][+][F][/][G][-][H][\*][I][+][J][\*][K][-][L][-][M][+][N][/][O][-][P]

printf("출력완료");

break;

case 3:

printf("\n후위순회 출력 : "); //[A][B][+][C][D][-][\*][E][F][+][G][H][-][/][+][I][J][+][K][L][-][\*][M][N][+][O][P][-][/][-][\*]

Postorder(n1);

printf("출력완료");

break;

case 4:

printf("\n모두 출력\n");

printf("\n전위순회 출력 : ");

Preorder(n1);

printf("\n중위순회 출력 : ");

Inorder(n1);

printf("\n후위순회 출력 : ");

Postorder(n1);

printf("출력완료");

break;

case 5:

printf("중위순회 순서로 주소값을 출력합니다 : \n");

Norder\_address(n1);

break;

case 6:

printf("\n모두출력 && 주소값 출력\n");

break;

default:

printf("Error");

break;

}

return 0;

}

**/\* 이진트리의 논리적구조**

**\***

**+ -**

**\* / \* /**

**+ - + - + - + -**

**A B C D E F G H I J K L M N O P**

**\*/**

int num1 = 0;

int num2 = 0;

int num3 = 0;

void Preorder(TreeNode\* root) { // 전위순회

num1++;

printf("Preorder %d번째 호출\n", num1);

if (root != NULL) {

printf("[%c]\n", root->data);

Preorder(root->leftchild);

Preorder(root->rightchild);

}

}

void Inorder(TreeNode\* root) { //중위순회

num2++;

printf("Inorder %d번째 호출\n", num2);

if (root != NULL) {

Inorder(root->leftchild);

printf("[%c]\n", root->data);

Inorder(root->rightchild);

}

}

void Postorder(TreeNode\* root) { //후위순회

num3++;

printf("Postorder %d번째 호출\n", num3);

if (root != NULL) {

Postorder(root->leftchild);

Postorder(root->rightchild);

printf("[%c]\n", root->data);

}

}

void Norder\_address(TreeNode \*root){

//주소값 출력하기!

if (root != NULL) {

Norder\_address(root->leftchild);

printf("\n[%C]의 데이터 주소 : %d\n", root->data, &root->data);

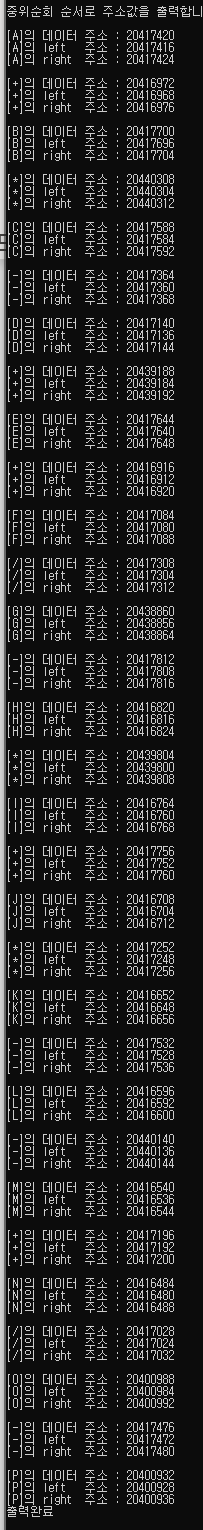
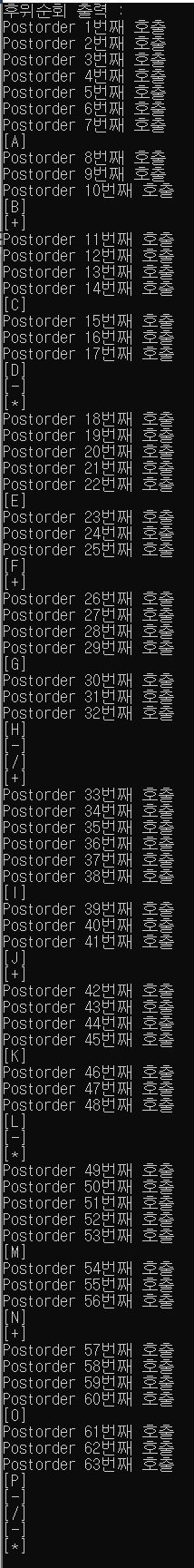
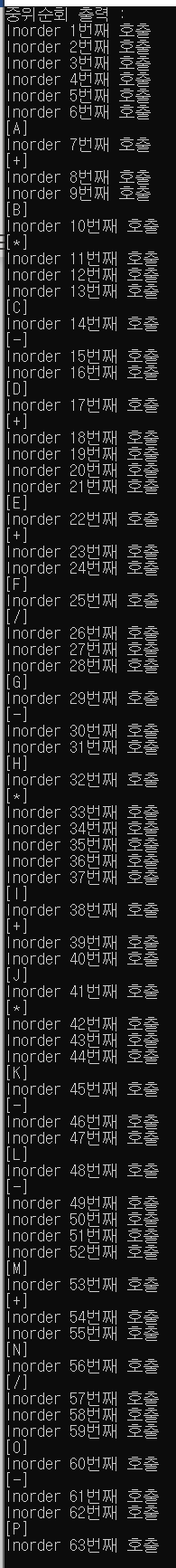
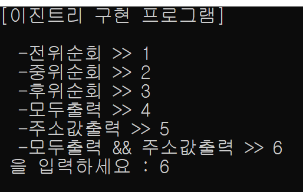
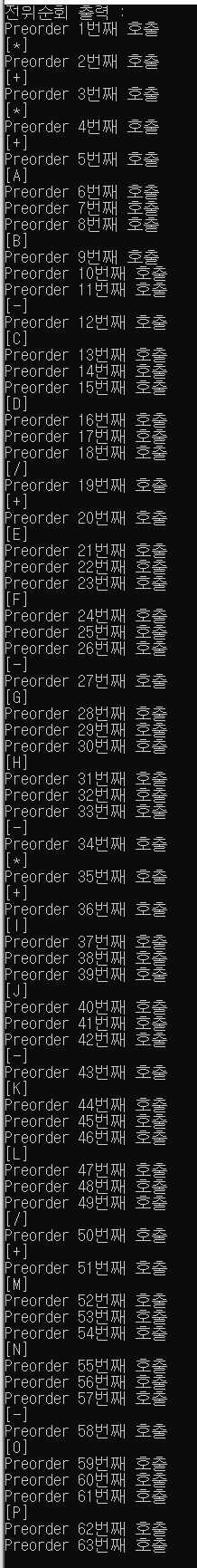
printf("[%C]의 left 주소 : %d\n", root->data, &root->leftchild);

printf("[%C]의 right 주소 : %d\n", root->data, &root->rightchild);

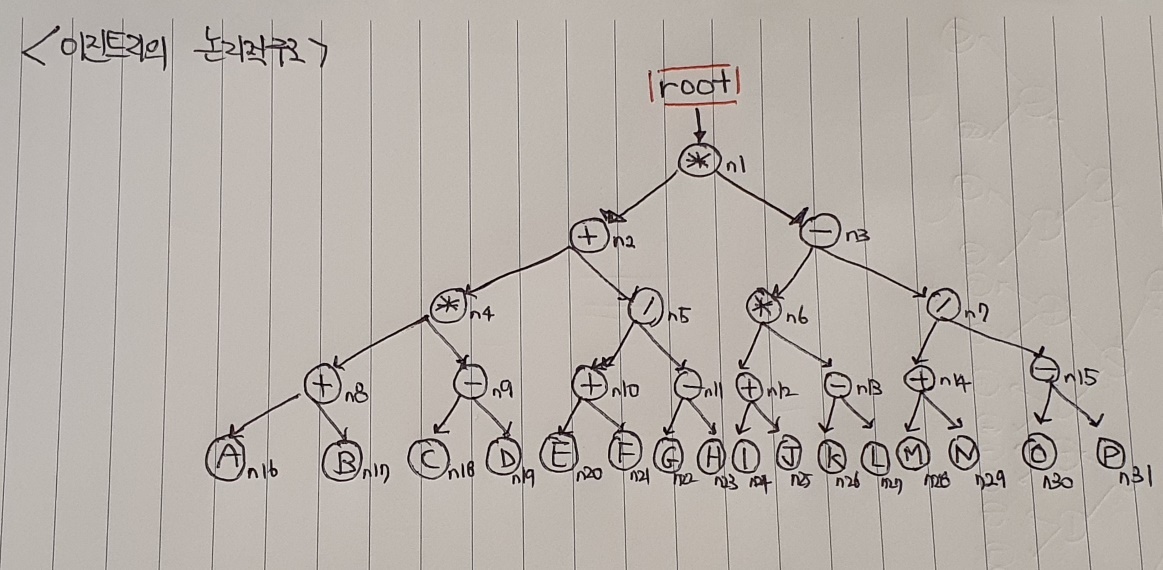
Norder\_address(root->rightchild);

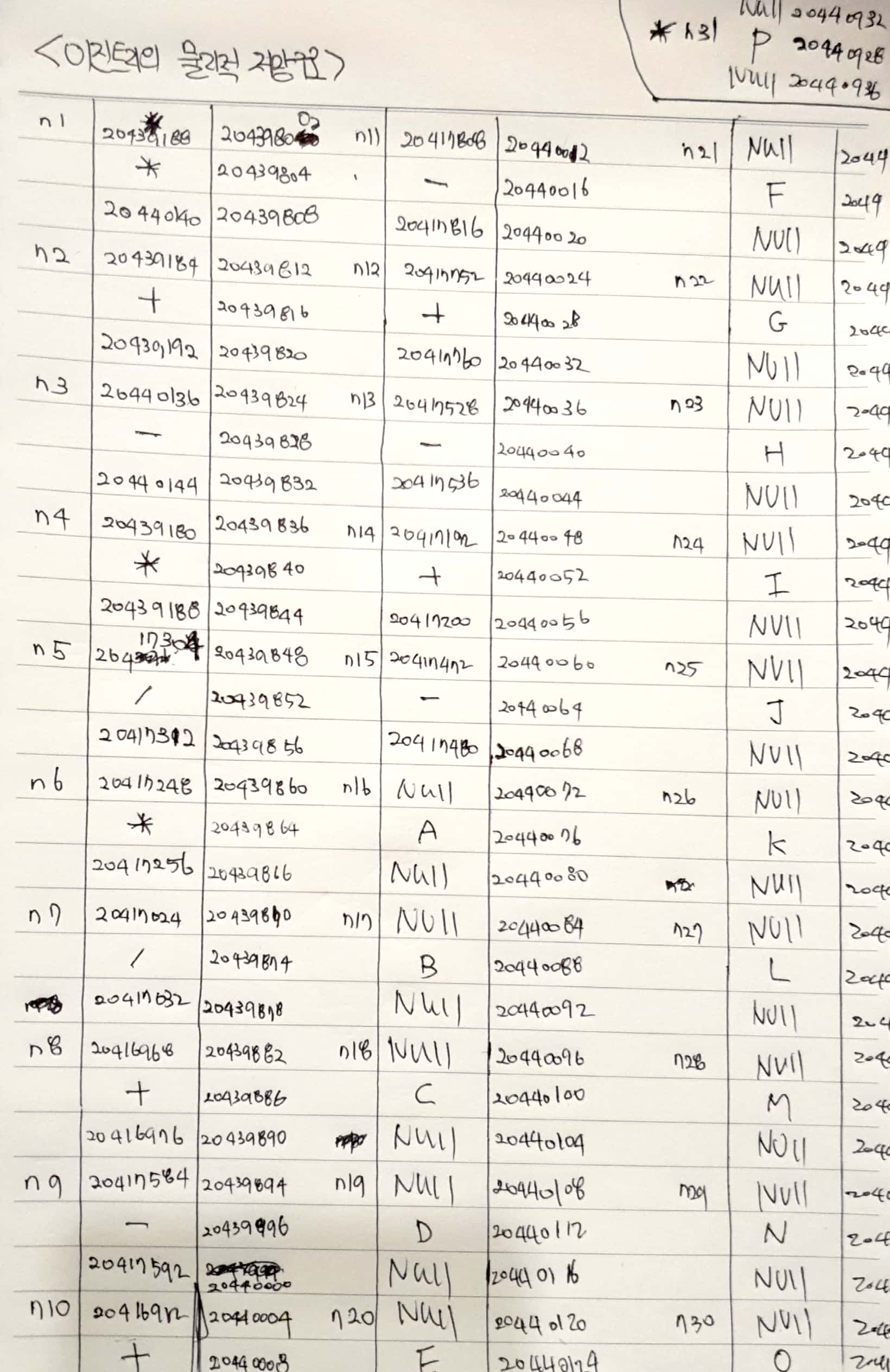
}

}

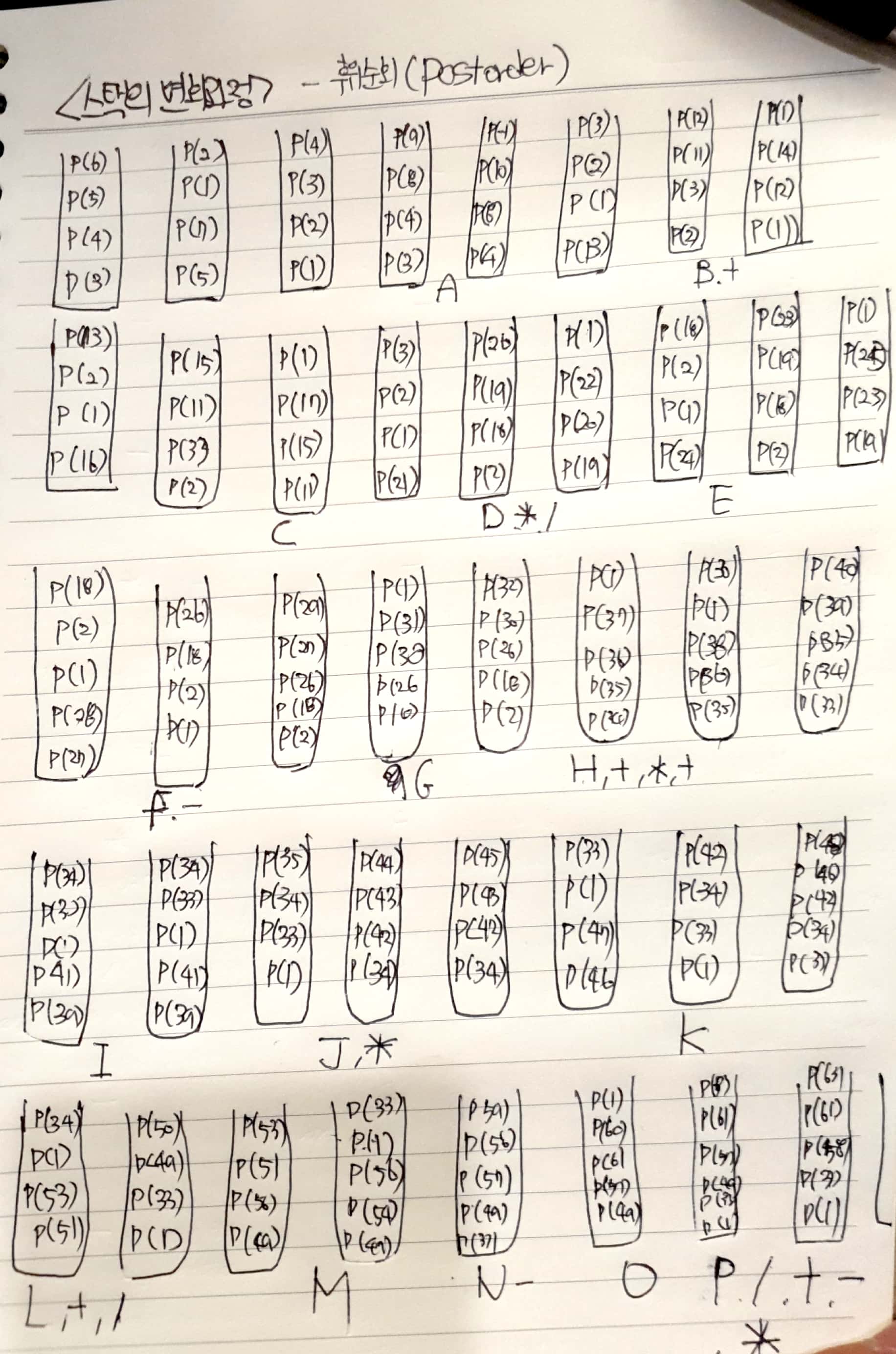
-실행스크린샷-[case6 = 모든순회 출력 && 주소값출력]

<논리적 구조>



<물리적구조>

<스택의 변화과정>



<트리를 구현하면서 에라가 발생했을 때 어떻게 수정했으며, 본 과제를 수행한 후의 소감>

안녕하세요 교수님

201910994 이정민 입니다.

교수님 수업을 들으면서 저번학기 C언어를 배울 때 어디에 적용하는건지 궁금함을 가지게 했던 구조체, 연결리스트, 배열, 동적메모리, 포인터 등의 함수들을 손에 익히는데 많이 도움이 되고 있습니다! 어떤 상황에서 어떻게 적용되고 사용되는지를 알고 나니까 이해도 잘되고 공부하는 재미가 붙고 있습니다!! ㅎ 항상 감사합니다.

연결리스트에 대한 이해가 다소 부족해서 실제 출력값은 순위순서와 달라서 코드를 여러번 수정하는 과정을 겪었습니다. 확실한건 이제는 연결리스트, 이진트리, 포인터는 정말 머리속에 정확히 이해했다는 점 입니다. 감사합니다 교수님!

수업 더욱 열심히 듣겠습니다.