4장: (head포인터가 리스트의 마지막을 가리키는) 원형 연결 리스트를 이용한 리스트 ADT(개정3판)

#include <stdio.h>

#include <stdlib.h>

typedef int element;

typedef struct ListNode { // 노드 타입

element data;

struct ListNode \*link;

} ListNode;

void print\_list(ListNode\* head) // 틀림(수정해야)

{

ListNode\* p;

if (head == NULL) return;

p = head->link;

do {

printf("%d->", p->data);

p = p->link;

} while (p != head);

printf("%d->", p->data); // 마지막 노드 출력

printf("\n");

}

ListNode\* insert\_first(ListNode\* head, element data)

{

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

node->data = data;

*if (head == NULL) {*

*head = node;*

*node->link = head;*

*}*

*else {*

*node->link = head->link; // (1)*

*head->link = node; // (2)*

*}*

return head; // 변경된 헤드 포인터를 반환한다.

}

ListNode\* insert\_last(ListNode\* head, element data)

{

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

node->data = data;

*if (head == NULL) {*

*head = node;*

*node->link = head;*

*}*

*else {*

*node->link = head->link; // (1)*

*head->link = node; // (2)*

*head = node; // (3)*

*}*

return head; // 변경된 헤드 포인터를 반환한다.

}

ListNode\* delete\_first(ListNode\* head)

{

ListNode \*temp;

if (head == NULL) {

printf("리스트가 비어 삭제를 못함\n");

return NULL;

}

*else if (head == head->link) { // 즉 하나의 노드가 남았을때*

*temp = head;*

*head = NULL;*

*free(temp);*

*}*

*else {*

*temp = head->link;*

*head->link = temp->link;*

*free(temp);*

*}*

return head;

}

ListNode\* delete\_last(ListNode\* head)

{

}

ListNode\* search(ListNode\* head, element data)

{

}

ListNode\* get\_size(ListNode\* head)

{

}

// 원형 연결 리스트 테스트 프로그램

int main(void)

{

ListNode \*head = NULL;

// list = 10->20->30->40

head = insert\_last(head, 20);

print\_list(head);

head = insert\_last(head, 30);

print\_list(head);

head = insert\_last(head, 40);

print\_list(head);

head = insert\_first(head, 10);

print\_list(head);

head = delete\_first(head);

print\_list(head);

/\*\*/

return 0;

}