자료구조 Scheduler

학번/이름 20131143최현

과목명 자료구조

지도교수 노승민 교수님

제출일 2014-06-12

**Source code**

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#define MAX\_LIST\_SIZE 100 //리스트크기 매크로

typedef struct ListNode{ // 스케줄의 date, time, data를 문자열로 저장함, 자기참조 구조체

char date[MAX\_LIST\_SIZE];

char time[MAX\_LIST\_SIZE];

char data[MAX\_LIST\_SIZE];

struct ListNode \*link;

}ListNode;

typedef struct{

ListNode \*head; //헤드포인터

int length; //노드의 개수

}ListType; //p151 리스트 ADT의 구현 참조

void init(ListType \*list)//리스트 초기화

{

if(list==NULL) return;//이미NULL이면 그냥 반환

list->length =0; //length값을 0으로 만듬(노드갯수초기화)

list->head = NULL;

}

int is\_empty(ListType \*list)//리스트가 비었나 확인

{

if(list->head == NULL)//head가 NULL을 가리키면 이미 비어있는 것이므로 1을 반환

return 1;

else //그외는 0반환

return 0;

}

void error(char \*message)//오류처리함수

{

fprintf(stderr,"%s\n",message);

exit(1);

}

ListNode \*get\_node\_at(ListType \*list, int pos) //pos위치의 노드를 일시적으로 tmp\_node에 저장하고 반환

{

int i;

ListNode \*tmp\_node=list->head;

if(pos<0) return NULL;

for(i=0;i<pos;i++)

tmp\_node=tmp\_node->link;

return tmp\_node;

}//p152 프로그램 4.22

void insert\_node(ListNode \*\*phead, ListNode \*p, ListNode \*new\_node) //노드 삽입과정

{

if(new\_node==(ListNode \*)NULL)

{

error("잘못입력하셨습니다.");

}

if(\*phead==NULL) // 공백리스트인 경우(삽입하려는 노드가 첫번째 노드)

{

new\_node->link=NULL;

\*phead = new\_node;

}

else if(p==NULL) // 선행노드를 가리키는 포인터 p가 NULL이면 리스트 맨 앞에 삽입

{

new\_node->link=\*phead;

\*phead = new\_node;

}

else //p 다음에 삽입

{

new\_node->link=p->link;

p->link=new\_node;

}

}//참고 교재 127p 프로그램4.5

void add(ListType \*list, int position, ListNode \*data1) // 주어진 위치에 노드 추가

{

ListNode \*p;

if((position>=0)&&(position <=list->length))

{

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

if(node==(ListNode \*)NULL)

{

error("메모리 할당에러");

}

node=data1;

p=get\_node\_at(list, position-1);

insert\_node(&(list->head), p, node);

list->length++;

}

}//p153 프로그램 4.22 참조

void remove\_node(ListNode \*\*phead, ListNode \*p, ListNode \*removed) // 노드 삭제 과정

{

if(p==NULL)

\*phead = (\*phead)->link;//p가 NULL이면 첫번째 노드 삭제 헤드포인터 변경 removed노드가 차지하고 있는 공간 반환

else

p->link = removed->link;//p가 removed 다음 노드 가리키게 변경

free(removed); //removed가 차지하던 공간을 시스템에 반환

}//p128 프로그램 4.6 참조

void is\_in\_list(ListType \*list, char a[]) // char a값과 리스트의 노드의 date값이 같으면 내용 출력

{

ListNode \*p;

p=list->head;

while(p!=NULL)

{

if(strcmp(p->date, a)==0) // p->date의 값(날짜)과 a값이 같으면

{

printf("%s %s %s\n", p->date, p->time, p->data);

p=p->link;

}

else

p=p->link;

}

}

//p155 프로그램 4.27 참조

void delete\_(ListType \*list, int pos) //주어진 위치의 노드 삭제

{

if(pos<0 || pos >= list->length)

{

error("위치 오류");

}

else if(!is\_empty(list) && (pos>=0) && (pos <= list->length)) // 리스트가 비어있지 않고 pos는 전체노드개수 보다 작고 0보다 커야한다.

{

ListNode \*p = get\_node\_at(list, pos-1); //pos-1번째의 노드를 p에 저장

remove\_node(&(list->head),p, (p!=NULL)?p->link : NULL);

// p가 NULL이 아니면 p->link, p가 NULL이면 NULL

list->length--;

}

}//p153 프로그램 4.23 참조

void clear(ListType \*list) //제일 처음에 있는 노드부분을 계속 삭제

{

int i;

for(i=0;i<list->length;i=0)

delete\_(list, i);

}//p155 프로그램 4.25를 참조

ListNode \*create\_node(ListNode \*link) // 빈 동적노드를 생성한 후 값을 받아서 저장하여 노드를 반환한다.

{

ListNode \*new\_node;

new\_node = (ListNode \*)malloc(sizeof(ListNode));

if(new\_node==NULL)

{

error("새 노드가NULL임");

}

else

{

printf("year/month/day hour:min memo : >>");

scanf("%s %s %s", new\_node->date, new\_node->time, new\_node->data);

new\_node->link = link;

}

return (new\_node);

}

void display(ListType \*list) //리스트 노드의 객체들의 값 표현

{

int i;

ListNode \*node=list->head; // 리스트 제일 앞의 노드주소를 node에 저장

for(i=0;i<list->length;i++) // 리스트 노드의 개수전까지 for문 수행

{

printf("(%d) %s %s %s\n", i, node->date, node->time, node->data);

node=node->link; //다음노드 진행

}

}//p155 프로그램 4.26 참조

int main(void)

{

ListType list1; //list1이라는 리스트를 생성한 후 여기에 노드를 추가한다

init(&list1);

printf("\t\t스케줄 관리 프로그램");

printf("\n\n도움말을 원하시면 h를 눌러주세요\n");

for(;1;)

{

char k1=0;

printf(">>");

scanf("%s", &k1);

if(k1=='h')

{

printf("i : 스케줄 입력\n");

printf("d : 스케줄 삭제\n");

printf("D : 모든 스케줄 삭제\n");

printf("p : 스케줄 확인\n");

printf("P : 모든 스케줄 확인\n");

printf("m : 스케줄 수정\n");

printf("q : 프로그램 종료\n");

printf("h : 도움말\n");

}

else if(k1=='i')

{

add(&list1, 0, create\_node(NULL)); //리스트의 처음에 새로운 노드 삽입

}

else if(k1=='d')

{

int n;

display(&list1);

printf("삭제하실 리스트의 번호를 입력하세요\n");

scanf("%d", &n);

delete\_(&list1, n);

printf("%d번이 삭제되었습니다.\n", n);

display(&list1);

}

else if(k1=='D')

{

char k2;

printf("정말로 모두 삭제하시겠습니까?(y/n)\n");

scanf("%s",&k2);

if(k2=='y' || k2=='Y')

{

clear(&list1); //y, Y를 누르면 모든 데이터 삭제

printf("모든 데이터가 삭제되었습니다.\n");

}

else if(k2=='n' || k2=='N')

{

printf("삭제를 취소합니다. \n");

}

}

else if(k1=='p')

{

char a[100]; // 리스트노드의 date와 동일한 문자열 생성

printf("확인할 스케줄 날짜를 입력하시오 \nyear/month/day>>");

scanf("%s", a);

is\_in\_list(&list1, a);

}

else if(k1=='P')

{

display(&list1); //모든 노드의 값을 보여준다

}

else if(k1=='m') // get\_node\_at은 x번째의 노드를 반환하기 때문에 ->를 쓸 수 있다

{

int x;

printf("리스트 번호를 입력하세요\n");

display(&list1);

scanf("\n%d", &x);

printf("수정전 : %s %s %s\n", get\_node\_at(&list1, x)->date, get\_node\_at(&list1, x)->time, get\_node\_at(&list1, x)->data);

delete\_(&list1, x);

add(&list1, x, create\_node(NULL));

printf("수정후 : ");

printf("%s %s %s\n", get\_node\_at(&list1, x)->date, get\_node\_at(&list1, x)->time, get\_node\_at(&list1, x)->data);

}

else if(k1=='q')

{

exit(1);

}

else

printf("잘못입력하셨습니다.\n");

}

return 0;

}

**문제 기술**

연결리스트를 이용해 스케줄러 만들기

**ADT 설계**

1. Insert()
2. Delete()
3. Delete\_all()
4. Display()
5. Display\_all()
6. Modify()
7. Quit()
8. Help()

**Function별 설명**

1. **Get\_node\_at (책 152p 프로그램 4.22)**

매개 변수 pos로 위치를 받아서 그 위치에 해당하는 노드의 주소를 반환한다.

Pos가 음수이면 NULL을 반환

Add와 delete modify에서 사용한다.

1. **Insert (책 127p 프로그램 4.5)**

Head가 NULL인 경우 > 삽입하려는 노드가 첫 번째 노드가 되므로 head의 값만 변경해줌

head

NULL

new\_node

P가 NULL인 경우 > 리스트의 맨 앞에 삽입 new\_node의 link가 head와 같은 값을 갖게 하고 그 다음에 head가 new\_node를 가리키게 함

head

NULL

new\_node

NULL

Head와 p가 NULL이 아닌 경우 > new\_node의 link에 p->link 값을 복사한 다음, p->link가 new\_node를 가리키게 함

p

Head

NULL

new\_node

NULL

1. **Add (책 p153 프로그램 4.22)**

주어진 위치에 노드를 추가

동적 메모리 할당으로 노드를 생성한 다음, 데이터 필드에 자료를 복사하고, 기존의 노드 중에 지정된 노드를 찾아 그 뒤에 연결 해줌

Get\_node\_at함수로 pos위치츨 받아 insert\_node 함수를 호출해 삽입을 수행함

1. **Remove\_node (책 p128 프로그램 4.6)**

노드를 삭제하는 과정이다

P가 NULL 일 경우 > 첫번째 노드를 삭제한다. 헤드포인터를 변경하고 제거 대상 노드가 차지하고 있던 공간을 시스템에 반환해준다.

head

NULL

removed

P가 NULL이 아닐 경우 > 제거 대상의 선행노드인 p의 링크가 removed 다음 노드를 가리키도록 변경한다.

p

Head

removed

1. **Is\_in\_list (책 p155 프로그램 4.27)**

모든 노드를 검색하여 데이터 필드가 char a[]인 노드를 찾는다

헤드포인터부터 NULL까지 while루프를 수행한다.

While 루프 안에서 strcmp를 이용해 date를 비교한다.

1. **Delete (책 p153 프로그램 4.23)**

공백이 아닌 리스트에서 pos위치의 자료를 삭제함

Get\_node\_at 함수로 (pos-1)위치의 노드 주소(노드의 선행노드의 주소)를 얻고

Remove\_node를 호출하여 노드를 삭제한다.

1. **Clear (책 p154 프로그램 4.25)**

모든 노드를 삭제하는 함수

책 소스를 그대로 가져다 놓을 시 전체삭제가 되지 않고 홀수 번 째 데이터만 지워졌다.

\*p는 1씩 증가하고 지워진 노드의 뒤에 있는 노드들은 앞으로 당겨지기 떄문에 지워지지 않는 노드가 생겼다. 이것은 for문에서 i를 증가시키지 않게끔 해 계속 맨 처음 노드만 가리키게 해서 전부 지워지게 하였다

1. **Create\_node**

빈 동적노드를 생성한 후 값을 받아서 저장하여 노드를 반환한다.

날짜, 시간, 정보를 입력 받기 위해 만들었다

1. **Display (책 p155 프로그램 4.26)**

리스트가 가지고 있는 정보를 화면에 출력하는 함수

List의 헤드포인터가 가리키는 노드에서부터 NULL을 만날 때 까지 노드의 데이터 값을 출력함

1. **Main**

메인에선 키를 입력 받은 대로 기능을 실행한다.

우선 list1이라는 리스트를 생성하고 이곳에 노드를 추가한다

i를 입력받으면 create\_node를 호출해 새로운 리스트에 새로운 노드를 추가한다.

P를 입력받으면 display를 호출해 모든 노드의 값을 보여준다

D를 입력받으면 한번더 되묻고 전체 데이터를 지운다

d 를 입력받으면 삭제할 리스트의 번호를 입력받아 그 번호에 해당하는 줄을 지우고 남은 노드값들을 보여준다

p를 입력받으면 리스트노드의 date와 똑 같은 문자열을 생성한다

여기서 확인할 스케줄 날짜를 입력해서 is\_in\_list 와 비교해 같은 값의 정보를 보여준다

m을 입력받으면 모든 노드값을 보여주고 원하는 리스트 번호를 입력받아 수정전을 보여주고 원래 노드를 삭제하고 다시 새로운 정보를 추가해 수정후도 보여준다.