강의명: 자료구조

숙제 번호: 02

숙제 제목: ADT Set(ADT 집합)

학생 이름: 김성현

학번: 201910783

1. ADT Set(집합)

1.1

s1910783@oak:^$ cp -r /home/ds/hw02 ./ds

s1910783@oak:ds$ ls

hw01 hw02

s1910783@oak:ds$ cd hw02

s1910783@oak:hw02$ ls

Makefile set set.c set.h

1.2

**<set.c>**

// ======================================================================

#include <stdio.h>

#include <stdlib.h>

#include "set.h"

// ======================================================================

SET \*create(void)

{

SET \*s;

s = malloc(sizeof(SET));

s->length = 0;

return s;

}

// ======================================================================

int member(SET \* set, int element)

{

int i;

for (i = 0; i < set->length; i++)

if (set->elements[i] == element)

return TRUE;

return FALSE;

}

// ======================================================================

int length(SET \* set)

{

return set->length;

}

// ======================================================================

SET \*insert(SET \* set, int element)

{

int i, j;

for (j = 0; j < set->length; j++)

if (element == set->elements[j])

return set;

for (i = 0; i <= set->length; i++) {

if (set->length == i)

{

set->elements[i] = element;

set->length++; //지금 length는 배열에 넣는 것을 위해 실제 길이보다 1작

break;

}

}

return set;

}

// ======================================================================

SET \*delete(SET \* set, int element)

{

int i, j;

for (i = 0; i < set->length; i++)

{

if (set->elements[i] == element)

{

for (j = i; j < set->length; j++)

{

set->elements[j] = set->elements[j+1];

}

set->length--;

}

}

return set;

}

// ======================================================================

SET \*sunion(SET \* set1, SET \* set2)

{

SET \*temp;

temp = create();

int i, j;

for (i = 0; i < set1->length; i++)

{

temp = insert(temp, set1->elements[i]);

}

for (j = 0; j < set2->length; j++)

{

temp = insert(temp, set2->elements[j]);

}

return temp;

}

// ======================================================================

SET \*intersection(SET \* set1, SET \* set2)

{

SET \*res;

res = create();

int i, e;

for (i = 0; i < set1->length; i++)

{

for (e = 0; e < set2->length; e++)

{

if (set1->elements[i] == set2->elements[e])

{

res->elements[res->length] = set1->elements[i];

res->length++; //길이를 맞게 증가.

break;

}

}

}

return res;

}

// ======================================================================

SET \*difference(SET \* set1, SET \* set2)

{

SET \*tem;

tem = create();

SET \*com;

com = create();

tem = sunion(set1, set2);

int i;

for (i = 0; i < set2->length; i++)

{

com = delete(tem, set2->elements[i]);

}

return com;

}

// ======================================================================

void print(SET \* set)

{

printf("{");

for (int i = 0; i < set->length; i++)

{

printf("%d", set->elements[i]);

if (i == (set->length-1))

break;

printf(", ");

}

printf("}");

printf("\n");

}

// ======================================================================

**프로그램을 이렇게 작성한 이유:**

1. SET \*create(void)

SET \*create(void)

{

SET \*s; //집합 생성을 위해 SET 구조체타입 s를 선언합니다.

s = malloc(sizeof(SET)); //하나의 집합은 최대 1024개까지 원소를 저장하기에 SET //타입의 크기 만큼 메모리를 할당하여 배열을 만듭니다.

s->length = 0; //아직 빈 배열이기 때문에 길이는 0으로 선언합니다.

return s; //이렇게 만든 s를 전달하기 위해 return 합니다.

}

=======================================================================

1. int member(SET \*set, int elements)

int member(SET \* set, int element)

{

int i; //for문 설정을 위해 i를 선언합니다.

for (i = 0; i < set->length; i++) //element가 포함되어있는지 확인을 위해 전체 배열을 탐색합니다.

if (set->elements[i] == element) //만약에 element와 같은 배열요소가 있을 경우를 조건으로 넣기 //위해 set->elements[i]번째에서 element를 비교시킵니다.

return TRUE; //포함되었다는 걸 알려주기 위해 TRUE를 반환합니다.

return FALSE; //위의 if문, 즉 배열 요소에 element가 포함되어 있지 않은 것을 반환하기 위해 for문이 완전히 끝난 후 FALSE를 반환합니다.

}

===================================================================

1. int length(SET \*set)

int length(SET \* set)

{

return set->length; //길이 반환을 위해 들어온 구조체형 set의 length 정수변수를 반환합니다.

}

===================================================================

1. SET \*insert(SET \*set, int element)

SET \*insert(SET \* set, int element)

{

int i, j; //for문 반복을 위해 변수 I, j를 선언합니다.

for (j = 0; j < set->length; j++) //추가하기 전 element가 배열변수 사이에 있는지 확인하기 위한 //for문.

if (element == set->elements[j]) //element가 set의 배열변수에 있는지if와 ==로 파악합니다.

return set; //만약 같은게 있는 경우 추가하는 것을 막기 위해 그대로 set을 반환합니다.

for (i = 0; i <= set->length; i++) { //원소 추가를 위해 for문을 선언합니다.

if (set->length == i) //배열 끝의 다음 자리에 왔을 때를 잡기 위해 i가 length랑 같아질 때를 //if문으로 둡니다.

{

set->elements[i] = element; //배열 끝의 다음의 빈자리에 element를 넣어 배열에 //추가되도록 합니다.

set->length++; //지금 length는 방금 넣은 자리 그 전까지, 즉 실제 길이보다 1작기에

//++을 통해 1을 증가해 추가된 element를 포함한 길이로 만들어줍니다.

break; //같은 걸 찾으면 더 이상 추가를 안하기 위해 break해줍니다.

}

}

return set; //이렇게 element가 추가된 set을 다시 주기위해 반환합니다.

}

========================================================================

1. SET \*delete(SET \*set, int element)

SET \*delete(SET \* set, int element)

{

int i, j;

for (i = 0; i < set->length; i++) //set 전체 탐색을 위해 for문을 선언합니다.

{

if (set->elements[i] == element) //배열원소가 element와 같을 때를 찾기 위해if문 //조건에 이 두개 같은 경우를 넣습니다.

{

for (j = i; j < set->length; j++) //같을 경우 그 배열요소 부분부터 for문으로 그 부분을 없애고 뒷배열을 앞으로 1칸씩 당기기 위해 for문 선언.

{

set->elements[j] = set->elements[j+1]; //앞 배열에 뒷배열의 값을 넣는 루프 //수행을 위해 이렇게 j, j+1를 넣어수행시킵니다.

}

set->length--; //앞으로 당기는 걸 끝내면 1개 줄어들었으니 길이를 1줄입니다.

}

}

return set; //이렇게 element가 delete된 set을 반환시킵니다.

}

=========================================================================

1. SET \*sunion(SET \*set1, SET \*set2)

SET \*sunion(SET \* set1, SET \* set2)

{

SET \*temp;

temp = create(); //합집합 저장을 위해 새로운 구조체형 temp을 선언합니다.

int i, j;

for (i = 0; i < set1->length; i++) //temp에 set1의 배열을 넣기위해 for문 사용.

{

temp = insert(temp, set1->elements[i]); //중복을 막기 위해 insert함수를 이용합니다.

}

for (j = 0; j < set2->length; j++)//그 다음 temp에set2의 배열 넣기 위해 for문 사용

{

temp = insert(temp, set2->elements[j]); //중복을 막기 위해 insert함수를 이용합니다.

}

//insert함수에 이미 중복되는 요소는 들어갈 수 없도록 처리가 되있기에 따로 교집합 구할 //필요없이 그냥 두 집합배열을 넣어주면 됩니다.

return temp;

}

======================================================================

1. SET \*intersection(SET \*set1, SET \*set2)

SET \*intersection(SET \* set1, SET \* set2)

{

SET \*res;

res = create(); //교집합 저장시키기 위해 새로운 구조체형을 선언합니다.

int i, e;

for (i = 0; i < set1->length; i++) //set1 탐색을 위한 for문

{

for (e = 0; e < set2->length; e++)//set1탐색 중 set2와의 비교를 위한 for문

{

if (set1->elements[i] == set2->elements[e]) //두 배열의 원소가 같을 경우

{

res->elements[res->length] = set1->elements[i]; //insert처럼 원소의 끝 //다음 자리에 교집합에 해당하는 원소를 넣기 위해 res->length를 씁니다.

res->length++; //하나 넣고 길이는 그대로이니 하나 늘려 새로 넣은 것까지 읽게 합니다.

break; //다 했으면 다음 set1의 원소와 set2 비교를 위해 break합니다.

}

}

}

return res; //이렇게 만들어진 교집합 res를 반환시킵니다.

}

=======================================================================

1. SET \*difference(SET \*set1, SET \*set2)

SET \*difference(SET \* set1, SET \* set2)

{

SET \*tem;

tem = create(); //합집합 저장을 위한 구조체형을 선언합니다.

SET \*com;

com = create(); //차집합 저장을 위한 구조체형을 선언합니다.

//차집합을 구하기 위해 A-B = (AUB) – B를 이용할 것입니다.

tem = sunion(set1, set2); //AUB를 위해 set1,set2의 합집합을 sunion함수로 tem에 저장합니다.

int i;

for (i = 0; i < set2->length; i++) //합집합 요소들을 비교하면서 빼기 위한 for문

{

com = delete(tem, set2->elements[i]);

//(AUB)-B를 위해 B에 해당하는 set2요소들을 delete함수를 이용해 빼서 com에 저장시킵니다.

}

return com; // 이렇게 구한 차집합을 주기 귀해 반환합니다.

}

=======================================================================

1. Void print(SET \*set)

void print(SET \* set)

{

printf("{"); //집합의 시작 중괄호를 따로 주기 위해 따로 printf시킵니다.

for (int i = 0; i < set->length; i++) //배열의 전체요소 출력을 위해 for문을 사용.

{

printf("%d", set->elements[i]); //배열의 요소 부분만 따로 출력하기 위해 printf.

if (i == (set->length-1)) //배열의 요소가 끝에 해당하는 [길이-1]에 다다르면 끝냅니다.

break;

printf(", "); //다음 요소를 출력하기 전 , 띄워쓰기를 위해 for문 끝나기전에 printf를 합니다.

}

printf("}"); //마지막으로 중괄호를 닫아주기 위해 printf로 출력합니다.

printf("\n"); //최종 마무리로 다음 줄로 넘기기위해 printf-\n을 사용합니다.

}

1.3

**수행결과화면:**

s1910783@oak:hw02$ gcc set.c -o set

s1910783@oak:hw02$ ./set

s1={-10, 30, -20, 5, 90, -70}

s2={5, -15, -20, -50, 15, -70, 90, -3}

member(s1,30)=TRUE

member(s2,-90)=FALSE

length(s1)=6

length(s2)=8

s1Us2={-10, 30, -20, 5, 90, -70, -15, -50, 15, -3}

s1^s2={-20, 5, 90, -70}

s1-s2={-10, 30}

s2-s1={-15, -50, 15, -3}

끝.