|  |  |
| --- | --- |
| sym01_s | **윈도우프로그래밍**  **LAB 05** |
| **분반** | **001** |
| **학번** | **2010044** |
| **이름** | **박진희** |

**# 실습 문제 1**

#1 Set.cpp 프로그램 소스 및 설명(수정한 결과 소스)

#include <iostream>

using namespace std;

#include "Set.h"

// default constructor creates an empty set

Set::Set(void):size(0),elems(){} //기본생성자

// array copy function, local to this file

static void array\_copy(int a[], const int b[], int n)

{

for (int i = 0; i < n; i++)

a[i] = b[i];

}

// copy constructor

Set::Set(const Set& s) //깊은 복사를 위해서 const 형식을 매개변수로 가져와야 한다.

{

this->size = s.size; //얕은복사와 달리 깊은복사는 모든 멤버값을 가져와야 한다. size 값을 가져온다.

array\_copy(this->elems, s.elems, size); //elems 배열을 가져온다.

}

// creates a set from an array

Set::Set(int a[], int n)

{

size = n; //n의 크기를 size에 대입

array\_copy(elems, a, n); //elems 배열에 a 배열을 n의 크기만큼 복사한다.

}

// return set size

int Set::getsize()

{

return size;

}

// return the element indexed by i; the lowest index is 1

int Set::get(int i)

{

if (i < 1 || i > size) {

cerr << "error in Set::get(): index out of bounds" << endl;

cerr << "returning 0" << endl;

return 0;

}

else

return elems[i - 1];

}

// set union

Set Set::operator+(Set s)

{

int i, j, n;

int a[2 \* MAXSIZE];

n = size;

array\_copy(a, elems, n);

for (j = 0; j < s.size; j++) {

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

if (a[i] == s.elems[j])

break;

if (i == n)

a[n++] = s.elems[j];

}

return Set(a, n);

}

// set difference

Set Set::operator-(Set s)

{

bool check; //동일한 원소가 있는지 확인하기 위한 bool

int a[MAXSIZE]; //차집합의 결과를 받기 위한 배열 선언

int n = 0; //배열의 size를 위한 int형 변수 n 선언

for(int i=0; i<size;i++){//this의 사이즈까지 인덱스로 사용하는 int i를 증가시킴

check = true; //우선은 기본적으로 true 값을 두어, 동일한 원소가 없을 시에는 true값을 계속 이어감.

for (int j = 0; j < s.size; j++) {//Set s의 사이즈까지 인덱스로 사용하는 int j 를 증가시킴

if (s.elems[j] == elems[i]) //만약 동일한 원소가 있다면 check에 false 값을 줌/

check = false;

}

if (check) //만약 check가 false로 변했다면 차집합을 구성하는 a에 추가되지 않는다.

a[n++] =elems[i];

}

return Set(a, n); //this 에 대해서 s와 겹치지 않는 원소만이 추가된 a배열을 새로운 Set의 인스턴스로 만들고 리턴한다.

}

// set assignment; return the assigned set

Set Set::operator=(Set s)//연산자 오버로딩

{

size = s.size; //사이즈 복사

array\_copy(elems, s.elems, size); //배열 복사

return (\*this);//this 자체는 포인터이므로 Set형으로 반환 위해 \*this 반환

}

// subset relation test

int Set::operator>=(Set s) //부분집합을 구하는 것이기 때문에 집합 자기자신과 같은 집합이어도 된다.

{

Set a = s - \*this; //s에서 this를 차집합을 하였을때 size가 0이라면 원소가 없다는 의미로, s가 this 의 부분집합이라는 의미이다.

if (!a.size) //size가 0이면 부분집합이므로, !0=1

return 1;

else

return 0; //size가 0이 아니므로 부분집합이 아님.

}

// set equality test

int Set::operator==(Set s)

{

Set a = s - \*this;

Set b = \*this - s; //서로에 대한 차집합이 공집합이라면 같은 집합이다

if (a.size == b.size&&!a.size&&!b.size) //사이즈가 서로 같고 0이라면 같은 집합

return 1;

return 0;

}

// proper subset relation test

int Set::operator>(Set s)

{

Set a = s - \*this; //앞서 정의된 부분집합 판단 연산과 같으나, 진부분집합은 자기 자신과 같은건 제외해야한다. 따라서 서로 사이즈가 같다면, 서로 같은 집합(즉 자기자신)이므로 0을 리턴한다.

if (s.size != this->size) {

if (!a.size)

return 1;

else

return 0;

}

return 0;

}

// element membership test; return 1 if (\*this) contains x

int Set::operator>>(int x)

{

for (int i = 0; i < size; i++) //this의 원소를 탐색해 x와 같은 값의 원소가 있다면 1을 반환한다.

if (this->elems[i] == x)

return 1;

return 0;

}

#2 main.cpp 프로그램 소스 및 설명(수정한 결과 소스)

#include <iostream>

using namespace std;

#include "Set.h"

void setprint(Set s)

{

cout << "{";

for (int i = 1; i <= s.getsize(); i++)

cout << s.get(i) << ((i < s.getsize()) ? ", " : "");

cout << "}";

}

void main()

{

int a[] = { 3, 9, 5 };

int b[] = { 2, 4, 5, 8, 10 };

Set s1;

Set s2(a, 3);

Set s3 = s2; // copy constructor is used here

Set s4(b, 5);

cout << "SET UNION TEST" << endl;

s1=s3 + s4;

cout << "s3 = "; setprint(s3); cout << endl;

cout << "s4 = "; setprint(s4); cout << endl;

cout << "s3 + s4 = "; setprint(s1); cout << endl << endl;

// s1 = s4 - s3

cout << "SET DIFFERNCE TEST" << endl;

s1=s4 - s3;

cout << "s4 = "; setprint(s4); cout << endl;

cout << "s3 = "; setprint(s3); cout << endl;

cout << "s4 - s3= "; setprint(s1); cout << endl << endl;

// test for i = 1, 2, ..., 10 if i is in s4

cout << "SET MEMBERSHIP TEST" << endl;

cout << "s4 = "; setprint(s4); cout << endl;

for (int i = 1; i < 11; i++) { //1~10까지 >>연산자가 1값을반환하면 참으로, is in s4를 출력하고 0을 반환하면 거짓으로 is NOT in s4를 출력한다.

if (s4 >> i)

cout << i << " is in s4" << endl<<endl;

else

cout << i << " is NOT in s4" << endl << endl;

}

// test if s4 > s1

// if not print < s1 is NOT a proper subset of s4 >

cout << "PROPER SUBSET RELATION TEST" << endl;

cout << "s1 = "; setprint(s1); cout << endl;

cout << "s4 = "; setprint(s4); cout << endl;

if (s4 > s1)

cout << "s1 is a proper subset of s4" << endl << endl;

else

cout << " s1 is NOT a proper subset of s4" << endl << endl;

// test if s2 == s3

// if not print < s2 is NOT equal to s3 >

cout << "SET EQUALITY TEST" << endl;

cout << "s2 = "; setprint(s2); cout << endl;

cout << "s3 = "; setprint(s3); cout << endl;

if (s2 == s3)

cout << "s2 is equal to s3" << endl << endl;

else

cout << "s2 is NOT equal to s3" << endl << endl;

}

#3 실행 화면

