3주차 결과보고서

전공: 기계공학과 학년: 3학년 학번: 20191820 이름: 김형준

**1. Array 클래스**

Array 클래스는 자료구조 중 1차원 배열을 사용하며, 객체 생성시 필요한 만큼의 메모리를 동적할당 받아 사용하고, 소멸시 할당받은 메모리를 해제한다.

int \*data : 동적할당 받은 메모리의 주소를 저장한다.

(배열의 데이터는 동적할당 받은 메모리에 저장됨)

int len : 배열의 크기를 저장한다.

Array의 배열이 가질 수 있는 인덱스는 0~len-1까지이다.

Array::Array(int size) // Array 생성자

{

//사이즈를 확인하고 양수이면 new를 사용하여 배열 data를 할당, len값 초기화

if (size < 0) // 사이즈가 유효하지 않은 값일 때

{

cout << "invaild Array size!" << endl; // 에러메세지 출력

}

else // 사이즈가 유효한 값일 때 (0이상)

{

len = size; // len을 size값으로 초기화

data = new int[len];

/\*

data에 new 연산자를 사용해 len개의 int를 저장할 수 있는 크기의

메모리를 할당받고, 할당받은 메모리의 주소를 data에 저장한다.

\*/

}

}

Array의 생성자이다. 클래스 Array타입 인스턴스를 생성할 때 호출되며, 유효한 값의 size를 사용하여 호출했을 때, 크기가 size인 int타입 배열을 동적할당 받는다. 동적할당된 배열의 주소와 크기를 각각 data와 len에 저장한다.

Array::~Array()

{

// 소멸자; 할당된 메모리 해제

delete []data; // 할당된 메모리를 해제한다.

}

Array의 소멸자이다. 호출시 data에 할당된 메모리를 해제한다.

int Array::length() const

{

// 배열의 크기 리턴

return len; // len의 값을 리턴한다.

}

배열의 크기를 리턴하는 함수다. 호출시 len의 값을 리턴한다.

int& Array::operator[](int i) // 배열에 원소 삽입

{

static int tmp; // static 변수 tmp 선언

/\* 배열의 인덱스가 범위 내에 있으면 값 리턴,

그렇지 않으면 에러메세지 출력하고 tmp리턴 \*/

if (i >= 0 && i < len) // i의 값이 유효한 값일 경우 (index로 가질 수 있는 값)

{

return data[i]; // data[i]를 리턴한다.

}

else // i의 값이 유효하지 않은 값일 경우

{

cout << "Array bound error!" << endl; // 에러메세지 출력

return tmp; // tmp를 리턴한다.

}

}

배열에 원소를 삽입하는 함수다. 연산자 다중정의를 사용하여 구현되었으며,

Arr[i] = 123; 형식으로 사용한다.

리턴값에 참조자를 사용해 Lvalue에 위치한 Arr[i]에 Rvalue의 값을 대입할 수 있다.

유효하지 않은 index를 사용했을 경우 에러메세지를 출력후 tmp를 리턴한다.

(이 경우, tmp에 Rvalue가 대신 저장된다.)

int Array::operator[](int i) const // 배열의 원소값 반환

{

/\* 배열의 인덱스가 범위 내에 있으면 값을 리턴,

그렇지 않으면 에러메세지 출력하고 0을 리턴 \*/

if (i >= 0 && i < len) // i의 값이 유효한 값일 경우 (index로 가질 수 있는 값)

{

return data[i]; // data[i]의 값을 리턴한다.

}

else // i의 값이 유효하지 않은 값일 경우

{

cout << "Array bound error!" << endl; // 에러메세지 출력

return 0; // 0을 리턴한다.

}

}

배열의 원소값을 반환하는 함수다. 연산자 다중정의를 사용하여 구현되었으며,

n = Arr[i]; 형식으로 사용한다.

Rvalue에 위치한 Arr[i]의 값을 Lvalue에 저장한다.

유효하지 않은 index를 사용했을 경우 0을 리턴한다.

void Array::print() //배열의 모든 내용을 출력해주는 함수

{

int i; // 루프용 변수 선언

cout << "["; // ‘[‘ 출력

for (i = 0; i < len; i++) // 배열의 크기만큼 반복

cout << " " << data[i]; // 배열의 모든 원소의 값을 출력함 (‘ ‘로 구분함)

cout << "]"; // ‘]’ 출력

cout << endl; // 개행 (‘\n’ 출력)

}

배열의 0번 인덱스부터 마지막 원소(len-1)까지 순서대로 출력한다.

**2. RangeArray 클래스**

RangeArray 클래스는 Array 클래스와 마찬가지로 자료구조 중 1차원 배열을 사용하며, Array 클래스를 상속받아 구현된다. (Array와 같은 방식으로 동적할당된 배열을 사용함)

int low : 배열 인덱스의 최소값을 저장한다.

int high : 배열 인덱스의 최대값을 저장한다.

RangeArray의 배열이 가질 수 있는 인덱스는 low~high이다.

RangeArray::RangeArray(int i,int j):Array(j-i+1) // Array 생성자 호출

{

low = i; // low를 i로 초기화

high = j; // high를 j로 초기화

}

RangeArray의 생성자이다. RangeArray가 가질 수 있는 인덱스가 low~high이므로

이 경우 배열의 크기는 high-low+1이다.

따라서 Array(j-i+1)을 호출하여 크기가 j-i+1인 배열을 가지는 Array를 생성한다.

이후 low와 high에 각각 i와 j의 값을 저장한다.

RangeArray::~RangeArray() {}

RangeArray의 소멸자이다. RangeArray는 Array를 상속받은 자식 클래스이고, data에 할당된 메모리는 Array의 소멸자에서 해제되기 때문에 따로 RangeArray의 소멸자에서 추가해 줄 부분은 없다. (RangeArray 소멸자 호출시 자동으로 Array 소멸자도 이어서 호출됨)

int RangeArray::baseValue()

{

return low; // low의 값 리턴

}

RangeArray의 low값을 리턴하는 함수다.

int RangeArray::endValue()

{

return high; // high의 값 리턴

}

RangeArray의 high값을 리턴하는 함수다.

int& RangeArray::operator [](int i)

{

return Array::operator [](i-low); // i-low번째 Array 원소를 리턴함

/\*

RangeArray는 low~high의 인덱스를 가지므로, RangeArray의 i번째 원소는

Array에서는 i-low번째 원소에 해당한다.

따라서 int& Array::operator[](int i)를 호출할 때 i의 값으로 i-low를 사용한다.

\*/

}

배열에 원소를 삽입하는 함수다. 연산자 다중정의를 사용하여 구현되었으며,

사용방법은 Array와 동일하다.

(Array의 int& Array::operator[](int i) 함수를 사용해서 구현함)

int RangeArray::operator [](int i) const

{

return Array::operator [](i-low); // i-low번째 Array 원소의 값을 리턴함

/\*

RangeArray는 low~high의 인덱스를 가지므로, RangeArray의 i번째 원소는

Array에서는 i-low번째 원소에 해당한다.

따라서 int Array::operator[](int i)를 호출할 때 i의 값으로 i-low를 사용한다.

\*/

}

배열의 원소값을 반환하는 함수다. 연산자 다중정의를 사용하여 구현되었으며,

사용방법은 Array와 동일하다.

(Array의 int Array::operator[](int i) 함수를 사용해서 구현함)

**3. (숙제 문제) Str 클래스**

Str 클래스는 문자열을 저장하기 위한 클래스로써, 자료구조 중 1차원 배열을 사용한다.

객체 생성시 입력한 문자열의 길이 또는 배열의 크기만큼의 메모리를 동적 할당받아

str에 할당받은 메모리의 주소를, len에 문자열의 길이를 저장한다.

char\* str : 동적할당된 문자열 저장용 배열의 주소를 저장한다.

int len : 문자열의 길이를 저장한다. (null문자 미포함)

Str(int leng) : Str 클래스의 생성자이다.

크기가 leng+1인 char타입 배열을 동적 할당한 뒤

str과 len에 각각 할당받은 메모리의 주소와 leng을 저장한다.

배열의 0~leng-1번 원소에는 공백문자(‘ ‘)를 저장하고 leng번 원소에는 null문자(‘\0’)를 저장해 문자열을 초기화한다.

Str(char\* neyong) : Str 클래스의 생성자이다.

strlen 함수를 사용해 neyong의 길이를 구한다음 len에 그 값을 저장한다.

크기가 len+1인 char 타입 배열을 동적 할당한 뒤

str에 할당받은 메모리의 주소를 저장한다.

strcpy 함수를 사용해 neyong의 문자열을 str에 할당받은 메모리에 그대로 복사한다.

~Str() : Str 클래스의 소멸자이다.

delete 연산자를 사용해 str에 할당받은 메모리를 해제한다.

int length(void) : 문자열의 길이를 리턴하는 함수이다.

return len;으로 현재 배열에 저장되어있는 문자열의 길이를 리턴한다.

char\* contents(void) : 문자열의 내용(주소)를 리턴하는 함수이다.

return str;로 문자열이 저장되어있는 배열의 주소를 리턴한다.

int compare(class Str& a) : 객체 a의 문자열과 str에 저장되어있는 문자열을 비교하는 함수이다.

strcmp와 Str 클래스의 char\* contents(void) 함수를 사용하여 str과 객체 a의 문자열을 비교한 결과값을 리턴한다. (return strcmp(str, a.contents());)

int compare(char\* a) : 문자열 a와 str에 저장되어있는 문자열을 비교하는 함수이다.

strcmp함수를 사용하여 str과 문자열 a를 비교한 결과값을 리턴한다.

(return strcmp(str, a);)

void operator=(char\* a) : 문자열 a의 값을 대입하는 함수이다.

str에 할당되어있는 메모리를 해제한 후, 문자열 a의 길이를 strlen 함수를 사용해 구하고 len에 그 값을 저장한다.

크기가 len+1인 char 타입 배열을 동적 할당한 뒤

str에 할당받은 메모리의 주소를 저장한다.

strcpy 함수를 사용해 a의 문자열을 str에 할당받은 메모리에 그대로 복사한다.

void operator=(class Str& a) : 객체 a의 값을 대입하는 함수이다.

str에 할당되어있는 메모리를 해제한 후, Str 클래스의 int length(void) 함수를 사용하여

len에 객체 a의 len의 값을 저장한다.

크기가 len+1인 char 타입 배열을 동적 할당한 뒤

str에 할당받은 메모리의 주소를 저장한다.

strcpy와 Str 클래스의 char\* contents(void) 함수를 사용해 객체 a의 문자열을

str에 할당받은 메모리에 그대로 복사한다.