1. 목적

효율적인 프로그래밍을 위해, class의 상속 방법을 익혀본다.

2. 실습 프로그램의 자료구조 및 알고리즘과 프로그램 구조도

(1) Array.h 및 array.cpp

Array 클래스를 구현하였다. Array 클래스는 멤버 변수로 int* 타입의 data, int 타입의 len을 가지며 모두 protected 변수이다. data 변수는 배열을 가리키고 있는 변수이며 len 변수는 배열의 길이를 저장하고 있는 변수이다.

Array 클래스의 멤버 함수는 모두 public 함수이다. Array 클래스는 배열의 크기를 인자로 받는 constructor와 메모리를 해제하는 destructor를 가진다. 또한 protected 변수인 len에 접근하여 배열의 크기 값을 리턴해주는 length() 함수를 가진다. 그리고 연산자 []를 operator overloading하여 배열의 원소에 값을 대입하거나 배열의 원소를 참조할 수 있도록 lvalue 및 rvalue 용 함수를 가진다. 대입 혹은 참조 시에 배열의 인덱스가 범위를 벗어나면 에러 메세지를 출력하였다. 마지막으로 print() 함수를 통해 배열의 모든 원소를 출력할 수 있도록 했다.

(2) RangeArray 및 rangearray.cpp

RangeArray 클래스를 구현하고 Array 클래스를 상속받도록 하였다. RangeArray 클래스는 시작 인덱스와 마지막 인덱스를 나타낼 수 있도록 멤버 변수로 int 타입의 base와 end를 가졌으며, 모두 protected 변수이다.

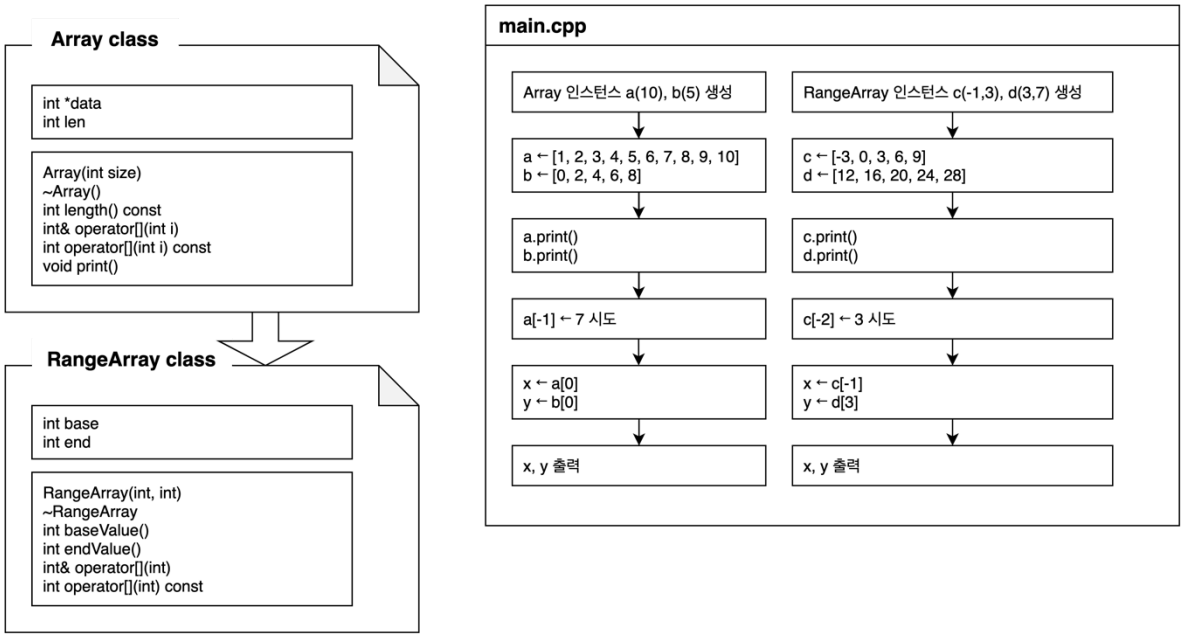
RangeArray 클래스의 멤버 함수는 모두 public 함수이다. RangeArray 클래스의 constructor는 Array 클래스의 constructor를 명시적으로 호출하였고 base와 end 변수에 전달받은 인자를 저장해주었다. RangeArray 클래스의 destructor는 비워두고 이후 Array 클래스의 destructor의 실행으로 메모리가 해제되도록 했다. 또한 baseValue(), endValue() 함수를 통해 시작 인덱스와 마지막 인덱스를 리턴하도록 했다. RangeArray 클래스에서도 연산자 []를 operator overloading하여 배열의 원소에 값을 대입하거나 배열의 원소를 참조할 수 있도록 lvalue 및 rvalue 용 함수를 가졌다. 이는 인덱스 조정을 위해 Array 클래스의 함수를 overriding한 것이다. Array 클래스를 상속받았기 때문에 배열의 모든 원소를 출력하는 함수는 별도로 작성하지 않았다.

(3) main.cpp

main 프로그램에서는 Array 인스턴스 a, b를 생성하고 배열에 값을 대입했으며, print() 함수를 이용해 그 내용을 출력하고, 인덱스가 범위를 벗어나도록 한 원소의 변경을 시도해보았다. 또 배열의 원소를 참조하여 int 타입의 x, y 변수에 대입한 뒤 그 값을 출력해보았다.

또한 RangeArray 인스턴스 c, d를 생성하고 마찬가지로 배열에 값을 대입했으며, print() 함수를 이용해 그 내용을 출력하고, 인덱스가 범위를 벗어나도록 한 원소의 변경을 시도해보았다. 또 배열의 원소를 참조하여 x, y 변수에 대입한 뒤 그 값을 출력해보았다.

(4) 프로그램 구조도



3. 과제 프로그램의 자료구조 및 알고리즘과 프로그램 구조도

(1) Str.h 및 str.cpp

Str 클래스를 구현하였다. Str 클래스는 멤버 변수로 char* 타입의 str, int 타입의 len을 가지며, 두 변수 모두 private 변수이다. str 변수는 문자열을 가리키고 있는 변수이며 len 변수는 문자열의 길이를 저장하고 있는 변수이다.

Str 클래스의 멤버 함수는 모두 public 함수이다. Str 클래스는 문자열 길이를 인자로 받는 constructor, 문자열 그 자체를 인자로 받는 constructor, 그리고 메모리를 해제하는 destructor를 가진다. 문자열의 길이를 인자로 받는 생성자의 경우 음수를 전달받으면 에러 메세지를 출력하고 프로그램을 종료한다. 또한 private 변수인 str, len에 접근할 때 사용하는 contents(), length()

함수를 구현했다. 각각 str과 len을 리턴한다. 그리고, 문자열의 비교에 사용되는 compare(class Str& a), compare(char *a) 함수를 구현하여 각각 Str 클래스의 객체와 문자열 비교에 및 char* 타입의 문자열과 비교가 가능하도록 하였다. 비교 결과 string.h의 strcmp 함수의 리턴 타입과 동일하게 두 문자열이 동일하다면 0을 리턴하고, 그렇지 않다면 음수/양수를 리턴하여 그 결과를 나타내주었다. 마지막으로 문자열의 수정 혹은 대입을 위해 연산자 =의 operator overloading을 했다. 이 경우에도 Str 클래스의 객체 또는 char* 타입을 전달 인자로 받는 함수를 각각 구현하여, 전달받은 문자열의 내용을 string.h의 strcpy 함수를 이용해 복사하여 str에 저장하고 len 변수의 값을 그에 맞게 바꿔주었다.

(2) main.cpp

main 프로그램에서는 Str 클래스의 a 인스턴스를 생성한 뒤, 멤버 함수 contents()를 이용해 내용을 출력해보았고, 연산자 =를 이용해 새로운 문자열을 대입하고 내용을 출력해보았다. 이후 멤버 함수 compare()를 이용해 "I'm a a" 문자열과 비교하여 그 결과를 출력해보았다.

(3) 프로그램 구조도