2020년도 2학기 컴퓨터공학설계및실험Ⅰ

2주차 UNIX-1: C/C++ Programming 예비보고서

20161663 허재성

1. 실습 목적

C++는 객체지향 프로그래밍 개념을 도입하여 클래스(Class)를 확장하고 수정하는 수단으로 클래스 상속(Class Inheritance)을 제공한다. 이번 실험에서는 효율적인 프로그래밍을 위하여 클래스의 상속 방법을 익혀보도록 한다.

2. 관련 이론

함수 다중정의(Overloading) : C++에서는 C와 다르게 이름이 같은 함수를 여러 개 정의할 수 있다. 함수의 이름이 같아도 함수의 인자의 개수나 자료형이 다르면 정의가 가능하다. C와 달리 C++에서는 함수를 좀 더 유연하게 사용할 수 있다.

참고로 함수의 반환형이 다른 것은 다중정의로 인정되지 않고 같은 함수를 중복 정의한 것으로 간주되어 에러가 발생한다.[1]

참조자 선언 : 참조자는 특정 변수를 가리키도록 하는 점에서 C언어의 포인터와 유사하지만 다른 개념으로 포인터 변수는 메모리 공간에 따로 할당되지만 참조자는 별도의 메모리를 할당하지 않고 변수와 함께 table 형태로 관리된다.[2]

|  |
| --- |
| int i; // 변수 i  int& p = i; // 변수 i를 가리키는 참조자 p 선언 |

변수 i와 i를 참조하는 참조자 p를 선언하면 p를 통해 i를 접근할 수 있다. 포인터 변수는 i를 가리키다가 다른 변수를 가리킬 수 있지만 참조자는 한 번 변수를 가리키면 다른 변수를 가리킬 수 없으며 선언과 동시에 초기화해야 한다. 이후의 Array 클래스를 구현할 때 [] 연산자를 다중정의하는 과정에서 배열의 원소에 접근하고 값을 대입할 수 있도록 참조자를 반환형으로 사용할 수 있다.

객체(Object), 클래스(Class)와 인스턴스(Instance) : 객체는 상태(state)와 행동(behavior)을 가지고 있는 모델로 클래스는 객체 지향 프로그래밍(OOP)의 특성을 표현하기 위한 C++ 문법으로 구조체의 속성과 유사하지만 더욱 확장된 형태이다. 클래스는 서로 관련이 있는 데이터를 멤버 변수로 묶어두고 이를 조작하기 위한 함수인 멤버 함수로 구성되어 있다. 클래스는 객체를 기술하는 틀(Template)라고 할 수 있으며 클래스를 통해 만들어진 객체를 인스턴스라고 한다.[3]

접근 권한 : 클래스의 멤버 변수와 멤버 함수를 클래스 외부에서 접근할 수 없도록 은닉할 것인지 여부를 결정한다. 일반적으로 외부에서 접근해서는 안되는 멤버 변수들은 접근 제어자 private로 선언하여 클래스 외에서 접근할 수 없도록 하며 멤버 함수들을 public으로 선언하여 외부의 객체가 클래스의 멤버 변수에 제한적으로 접근할 수 있도록 한다. 클래스와 해당 클래스를 기반 클래스로 상속받은 파생 클래스에서만 접근 가능하도록 protected로 선언할 수도 있다.

캡슐화(Encapsula), 정보 은닉(Information Hiding)

정보 은닉은 클래스 외부에서 클래스의 멤버에 불필요하게 접근하는 것을 막아 프로그램을 좀 더 견고하게 작성하는 것이다. 캡슐화는 클래스의 데이터를 멤버 변수로, 데이터를 조작하는 기능을 멤버함수로 하나로 묶는 것이다.

생성자(Constructor)와 소멸자(Destructor)

생성자는 객체가 처음 생성될 떄 무조건 호출이 되는 함수로 리턴값이 존재하지 않는다. 클래스에 생성자를 정의하지 않았을 경우 어떤 인자도 받지 않는 디폴트 생성자가 호출된다. 생성자도 함수이므로 다중정의가 가능하며 특정 인자를 받는 생성자를 정의하면 디폴트 생성자는 호출되지 않는다. 생성자의 이름은 클래스와 동일하다. 생성자는 클래스의 인스턴스를 생성할 때 무조건 한 번 호출되는 함수로 일반적으로 멤버 변수를 초기화하거나 메모리를 동적할당받는데 사용된다. 이후에 정의할 Array 클래스에서는 생성자에서 메모리를 동적할당할 것이다.

소멸자는 객체가 소멸될 때 호출되는 멤버 함수로 생성자와 같이 클래스의 이름을 그대로 가지며 어떤 인자도 반환값도 없다. 생성자와 구분하기 위해 이름 앞에 틸다(~)를 붙이며 일반적으로 클래스에서 동적할당받은 메모리를 해제하는 데 사용된다.

상속(Inheritance)

상속은 클래스가 인스턴스를 생성할 때, 다른 클래스의 멤버 변수/ 함수를 빌려와서 자신의 멤버 변수/함수와 합쳐 하나의 인스턴스를 생성하는 것이다. [4]  여러 클래스의 공통적인 개념을 추출하여 독립적인 클래스로 정의하고 이러한 공통적인 개념을 상속받고 각 클래스별로 필요에 따라 특수화하는 것이다. 공통적인 개념을 추상화한 클래스를 기반 클래스(Base Class)라고 하고 기반 클래스를 상속 받은 클래스를 파생 클래스(Derived Class)라고 한다. 파생 클래스는 기반 클래스를 상속받아 기반 클래스의 각각 속성, 기능에 해당하는 멤버 변수/함수에 자신의 멤버 변수/함수를 더하여 원래 자신의 것처럼 인스턴스를 생성한다. 이후에 Array 클래스를 정의하고 Array 클래스를 상속받는 파생 클래스 RangeArray 클래스를 정의한다.

함수 재정의(Overriding)

기반 클래스의 멤버 함수를 파생 클래스가 상속받아서 그대로 사용하는 것이 아니라 파생 클래스 내에서 다시 작성하는 것이다. 파생 클래스의 인스턴스를 생성 후 해당 함수를 실행하면 파생 클래스에서 재정의한 함수가 실행된다. 후에 실습할 RangeArray 클래스는 Array 클래스에서 다중정의한 [] 연산자들을 상속받아 그대로 사용하는 것이 아니라 RangeArray의 특성 및 기능에 맞게 재정의하여 사용한다.

연산자 다중정의(Operator Overloading)

일반적으로 기본 자료형인 int, double, char 등에서 사용되는 연산자(+, -, \*, /, … 등)에 다른작업을 할당해서 연산자가 복수의 의미를 가지도록 한다. C++에서 제공하는 기본 타입이 아닌 사용자가 임의로 정의한 클래스 타입에서도 연산자를 사용할 수 있도록 한다.[5] 연산자 다중정의를 통해 멤버 함수로 구현했을 때 보다 가독성을 높이고 사용자가 더 쉽게 사용할 수 있다. 후의 실습에서 1차원 배열을 구현한 Array 클래스와 RangeArray 클래스에서 [] 연산자를 다중정의해 기본 자료형 배열처럼 특정 인덱스에 접근할 수 있도록 한다.

3. 실습 방법

기본 자료구조인 int형 배열을 추상화한 클래스 Array를 먼저 구현해야 한다. 기본 자료구조인 배열과 마찬가지로 Array 클래스 또한 선언할 때 크기를 정해줘야 하고 Array의 특정 인덱스에 원소를 대입하거나 특정 인덱스의 원소를 반환 받을 수 있어야 한다. 기본 자료구조 배열과 달리 동적으로 메모리를 할당받으므로 Array 클래스의 인스턴스들은 사용이 끝났을 시 메모리를 해제해 주는 작업이 필요하다. 또한 기본적인 배열 자료구조가 자체적으로 제공하지 않는 기능인 배열 내용 전체를 출력해주는 기능을 Array 클래스의 멤버 함수로 정의해야 한다. 추가적으로 배열의 범위를 벗어나는 인덱스에 접근하여 값을 대입하려 하거나 참조하려 할 때 에러 메시지를 출력하고 비정상적으로 종료되거나 비정상적인 값을 반환하도록 한다.

1차원 배열을 동적할당하기 위해서 Array 클래스의 멤버변수로 int형 단일 포인터 변수 data와 배열의 크기를 나타낼 변수 정수형 변수 len이 필요하다. 이 때 멤버변수는 Array 클래스와 Array 클래스를 상속받는 클래스들을 제외하고 접근할 수 없도록 접근 지정자를 protected로 선언한다. Array 클래스의 인스턴스를 생성할 때 생성자 Array에 인자로 배열의 크기가 들어오면 크기가 유효한지 검사 후 유효하다면 동적할당을 한다. 만약 0 이하의 값으로 유효하지 않다면 에러 메시지를 출력 후 프로그램을 종료한다. 메모리를 동적으로 할당받은 뒤에는 배열의 연산처럼 특정 인덱스의 원소에 값을 대입하고, 참조할 수 있도록 [] 연산자를 다중정의해야 한다. 특정 인덱스의 원소(left value)에 값을 대입하기 위해서 연산의 결과가 int 참조형인 [] 연산자와, 인덱스의 원소의 값을 반환 받기 위해(right value) 연산의 결과가 int형인 [] 연산자를 다중정의한다. Int형 [] 연산자는 Array의 원소를 단순히 가져오기만 할 뿐, 값은 변경하지 않으므로 const 선언을 추가한다. 두 연산자 모두 범위를 벗어난 인덱스에 접근하면 오류 메시지를 출력하도록 작성한다.

멤버 함수로 배열의 크기를 클래스 외부에서 알 수 있도록 반환해주는 length() 함수를 정의한다. 마찬가지로 const 선언을 추가해준다. 마지막으로 동적 할당한 배열의 원소를 모두 출력해주는 print() 함수를 정의한다. 마지막으로 Array 클래스의 동적 할당된 메모리를 해제해주도록 소멸자 ~Array()를 정의한다.

RangeArray 함수는 Array 클래스를 기반 클래스로 상속받는 파생 클래스이다. Array 클래스의 멤버 변수는 protected로, 멤버 함수는 public으로 선언했으므로 멤버 변수와 멤버 함수, 연산자 다중정의를 그대로 이용할 수 있다. 기본적으로 인덱스가 0부터 시작하는 Array와 다르게 RangeArray는 배열의 시작 인덱스와 끝 인덱스를 선언할 때 지정해 줄 수 있도록 정의한다. 따라서 멤버 변수에 배열의 시작 인덱스와 끝 인덱스를 나타내는 base와 end를 private로 선언한다. RangeArray 클래스의 생성자는 Array 클래스의 생성자와 달리 두 개의 정수 인자를 받아 각각 base와 end로 한다. 배열의 크기를 계산한 후 그 값을 기반 클래스의 생성자를 호출하여 넘겨주면 인덱스가 0부터 시작하는 동적 배열이 할당된다. RangeArray에 알맞게 대입과 반환 연산을 하기 위해서 기반 클래스에서 다중 정의(overloading)한 [] 연산자를 재정의(overriding)해야 한다. RangeArray 클래스에서 새로 정의한 멤버 변수인 base와 end 변수를 이용하여 기반 클래스의 data의 대응하는 인덱스의 원소를 반환하도록 재정의해야 한다. 추가적인 멤버 함수로 클래스 외부에서 RangeArray로 구현한 배열의 시작 인덱스와 끝 인덱스를 알 수 있도록 반환하는 멤버 함수 baseValue()와 endValue()를 정의한다. 기반 클래스의 length() 함수와 print() 함수는 재정의할 필요없이 상속받은 함수를 그대로 사용할 수 있다.

4. 기타

이번 실습에서 사용하는 연산자 다중정의는 클래스의 멤버 함수처럼 클래스 내에 정의한다. 클래스 내에 정의하는 것이 아닌 클래스 밖의 전역 함수처럼 연산자를 다중정의할 수도 있다. 클래스 내에 연산자를 다중정의하면 연산자의 왼쪽 항이 클래스의 객체여야 한다.[6]

그 외에도 함수 호출 연산자 (), 메모리 접근, 클래스 멤버 접근 연산자 \*, ->, 형변환 연산자도 중정의할 수 있다. 연산자 다중정의는 STL을 이해하는데 도움이 된다. [7]

5. 참고 문헌

[1] 윤성우, “윤성우의 열혈 C++ 프로그래밍[개정판]”, *오렌지미디어,*  2010

[2][3][4] 서강대학교 컴퓨터공학과 교수진, "컴퓨터공학 설계 및 실험 I", 서강대학교 공학부 컴퓨터공학과, 2012

[5][6][7] 공동환, “뇌를 자극하는 C++ STL”, [*한빛미디어*](http://book.interpark.com/product/BookDisplay.do?_method=detail&sc.prdNo=210124199&gclid=EAIaIQobChMIzoWF7M_w6wIVBFZgCh1lrQQqEAQYASABEgJu__D_BwE), 2012