

ICE2004 자료구조론 과제1

제 목

객체지향 프로그래밍 H/W

보고서 작성 서약서

1. 나는 타학생의 보고서를 베끼거나 여러 보고서의 내용을 짜집기하지 않겠습니다.

2. 나는 보고서의 주요 내용을 인터넷사이트 등을 통해 얻지 않겠습니다.

3. 나는 보고서의 내용을 조작하지 않겠습니다.

4. 나는 보고서 작성에 참고한 문헌의 출처를 밝히겠습니다.

5. 나는 나의 보고서를 제출 전에 타학생에게 보여주지 않겠습니다.

나는 보고서 작성시 윤리에 어긋난 행동을 하지 않고 정보통신공학인으로서 나의 명예를 지킬 것을 맹세합니다.

2021년 10월 8 일

학부 정보통신공학과

학년 2

성명 김민겸

학번 12201863



1. **개요**

과제1은 객체지향 프로그래밍의 다양한 기능을 함축적으로 담은 소스 코드를 구현하는 것이다. 아래는 과제1에서 요구하는 기능의 주요 키워드이며, 키워드마다 해결해야 하는 과제의 세부 사항을 분석하여 기재하였다.

1. Virtual (다형성)

* 상속 관계 클래스의 멤버 함수 play()를 Virtual 기능을 사용하여, 객체마다 다른 출력문이 나오도록 처리한다.
* 소멸자에 virtual을 사용하여, new한 객체들을 delete 시 정상적으로 소멸자가 호출되도록 한다.

1. 연산자 오버로딩

* 출력 연산자 오버로딩
* + 연산자 오버로딩
* = 연산자 오버로딩

1. 클래스 상속

* 위 UML 다이어그램과 동일한 형태의 클래스 계층 구조를 구현.
* 생성자, 소멸자가 객체 생성 시 정상적으로 호출되도록 구현

1. Static 멤버 변수

* 객체의 개수를 의미하는 count 변수를 static으로 정의.

소스 코드의 큰 타이틀은 위와 같이 분류해 놓았지만, 정작 프로그램을 구현하면서 가장

신경 쓴 부분은 ‘포인터’ 부분이다.

1. **구현 상 특징**
2. 클래스 상속 관계

****

****

****

주어진 UML 다이어그램에 따르면 위의 사진처럼 세 가지 상속 관계가 주어져 있고, 모두 public 상속을 받았다. Public을 따로 작성하지 않으면 default 값인 ‘private 상속’이 이루어지게 되고, private 상속에서는 상위 클래스의 private 멤버에 접근할 수 없기 때문에 mp배열에 new 연산자를 이용하여 자식 클래스의 구현체를 만드는 과정에서 에러 사항이 발생한다.

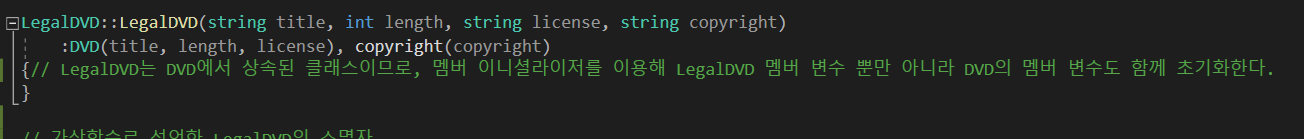
따라서 public으로 상속을 해주며, 각각의 생성자는 다음과 같이 구현하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명



텍스트이(가) 표시된 사진

자동 생성된 설명

보는 바와 같이, Media는 가장 상위 클래스이고 DVD, VTR, LegalDVD가 주어진 UML 다이어그램과 같이 상속 관계를 지니고 있으므로 멤버 이니셜라이저를 이용해, 각각의 자식 클래스의 생성자에서 부모 클래스의 생성자를 호출하여 연쇄적으로 값을 초기화할 수 있다.

1. 연산자 오버로딩





다음 코드를 구현하기 위해서, 각각 << 연산자 오버로딩과 +연산자 오버로딩을 수행할 필요성이 있다.

1. **+ 연산자 오버로딩**

**텍스트, 화면이(가) 표시된 사진

자동 생성된 설명**

+ 연산자 오버로딩 과정에서 고려해주어야 할 점은 반환형, 매개변수, 리턴하는 객체 총 세 가지이다. Shallow Copy가 이루어질 것을 고려해 +연산자 오버로딩의 결과로 리턴해야 하는 객체는 포인터 자료형이었고, 이에 따라 Media \*를 함수의 반환형에 작성하였다.

그리고 매개변수로는 mp[2], 즉 Media \* 자료형을 피연산자로 사용하고 있으므로 + 연산자 오버로딩의 매개변수에는 Media \* 자료형의 객체가 들어오게 된다. 이때 매개변수로 들어온 객체의 멤버 변수에는 변경의 여지가 없어야 하므로, const 형식으로 지정한다.

정의 내에서는 Media 포인터 형태의 temp 객체를 만들어서, new 연산자를 사용해 Media 클래스의 구현체를 만든다. 실제 구현체는 VTR 자료형이며, VTR 객체인 mp[3]과 Shallow Copy가 일어날 것을 대비한 것이다. 따라서 Title은 mp[3] 객체와 동일하게 작성하고, VTR의 매개변수 format역시 값을 동일하게 한다. Length는 현재 객체의 Length와 매개 변수로 들어온 Length값을 합하여 temp 변수의 length에 할당한다.

마지막으로, temp를 리턴해주면 이 주소값을 가져와서 mp[3]변수에 할당할 수 있다.

1. **<< 연산자 오버로딩**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

다른 연산자 오버로딩과는 다르게 <<연산자 오버로딩은 전역함수로 정의하였다. 이는 표준 라이브러리인 iostream에 정의되어 있는 클래스 ostream의 내부 멤버 변수에 접근할 수 없기 때문이다.

이때 매개변수는 ostream 객체와, 객체의 포인터 자료형이며, 정의 부분에서 play()함수를 호출하여 객체의 구현체에 맞는 출력이 나오도록 구현하였다. 그리고 output을 리턴해서 중복 출력이 가능하도록 구현했다.

1. Virtual 다형성 구현

play함수와, 자식 클래스의 소멸자에 대해서 virtual 기능을 사용하였다.

Virtual은 상속 관계에서 부모 클래스와 자식 클래스의 함수 이름이 동일할 때 오버라이드, 즉 자식 클래스의 함수를 재정의하기 위해서 사용하는 것이다. Driver.cpp 파일의 mp배열은 Media 포인터 자료형에 실제 구현체가 하위 클래스 객체를 담는 배열이다. 즉 자식 클래스의 멤버 함수가 호출되도록 구현하는, ‘다형성’ 기능을 활용하는 것이다.



이렇게 각각의 자식 클래스에 대해 play() 함수에 대해서, 각 구현체에 걸 맞는 출력문이 나타난다.



그리고 소멸자에도 동일하게 virtual 키워드를 사용하였는데, 그 이유는 다형성이 구현되어 있는 상황에서 객체를 생성했을 때 프로그램이 종료되고 상위 클래스인 Media 클래스에 대해서만 소멸자가 실행되기 때문이다. Virtual을 사용하면 자식 클래스의 소멸자까지 정상적으로 출력되게 된다.

1. Static 멤버 변수

클래스 내부의 static 멤버 변수는 객체의 생성과 무관하게 독립적으로 생기며, 모든 인스턴스에서 공동으로 접근할 수 있다. 이는 클래스 외부에서 전역 변수처럼 초기화되어야 하는 특징이 있다. 해당 프로그램에서는 Driver파일에서 변수를 불러와야 하므로 다음과 같이 작성하였다.



그리고 객체가 생성될 때마다 count의 값을 증가시켜주어야 하므로, 가장 상위 클래스인 Media의 생성자 부분에 아래 코드를 넣었다.

텍스트이(가) 표시된 사진

자동 생성된 설명

이렇게 되면 객체를 4개 생성한 후 Media::count의 값이 4가 되는 것을 확인할 수 있다.

**실행화면 캡쳐 이미지**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

1. **고찰**

+ 연산자 오버로딩을 하는 과정에서 this 포인터를 그대로 리턴하는 과정에서 Pointer Dangling 에러를 마주하게 되었다.

텍스트이(가) 표시된 사진

자동 생성된 설명

처음에 작성하였던 코드인데, mp[1] 현재 객체의 length값을 매개변수로 들어온 mp[2]의 length 값을 더해주어 객체를 그대로 리턴해주었다. 이런 상황에서 반환된 Media \* 객체, 즉 mp[1]을 담고 있는 this를 리턴하여 mp[3] 포인터 변수에 대입함으로써, mp[1]의 주솟값을 mp[3]의 포인터 변수에 담게 된다.



즉 위의 코드에서 대입 연산자에 의해 Shallow Copy가 일어나 mp[3] 포인터 변수가 mp[1] 객체를 동시에 가리키게 되어서, delete시 이미 delete된 mp[1]이 다시 delete되는 Pointer Dangling이 생긴다. 그래서 delete 과정에서 메모리 손상의 문제가 일어나는 것이다.

**텍스트, 화면이(가) 표시된 사진

자동 생성된 설명**

이러한 문제는 + 연산자 오버로딩 내부에서 return this; 가 아닌 새로운 temp 객체를 정의해줌으로써 해결하였고, 이 temp객체의 주솟값이 mp[3]에 대입되어, 업데이트된 length값이 담긴 temp 객체가 mp[3]의 객체가 된다.

텍스트, 실내, 화면, 은색이(가) 표시된 사진

자동 생성된 설명

처음에는 위 코드에서 = 연산자 오버로딩이 실행된다고 생각해서 return this;를 수행한 다음 = 연산자 오버로딩에서 매개변수에 대한 delete를 수행하면 포인터 연결을 끊어주어 정상적으로 실행될 것이라 생각했다.

그러나 mp[3] 객체의 앞부분에 \* 가 없으므로 애초에 오버로딩이 불가능해, Shallow Copy가 일어날 수밖에 없다는 것을 알고 temp 객체를 만들어주는 방법을 택하였다. (= 연산자 오버로딩 부분은 삭제하였다.)

이렇게 해서 length값을 update하는 목적을 달성할 수 있었으나, 다만 코드의 한계점은 Driver.cpp 파일에서 mp[3] 객체에 주솟값이 대입된다는 것을 미리 알고 있었으므로 + 연산자 오버로딩을 위와 같이 구현하여 객체 내부의 멤버 변수(title, format)이 바뀌지 않을 수 있는 것이다. 만일 mp[3]이 아니라 다른 객체에 temp가 대입된다고 하면, 그 다른 객체의 멤버 변수의 값이 바뀌는 버그가 생기게 된다.

이렇게 멤버 변수를 유지할 수 없다는 단점이 있지만 length값이 더해지는 구현 상의 목적을 달성하였으므로 오버로딩을 잘 구현했다고 생각한다.