학번	작성자
20131575	유승재

파일처리 프로젝트 #1

인터넷 강의 회원관리 시스템

'2018. 10. 09

차 례

서강대학교 컴퓨터공학과

서강대학교 SOGANG UNIVERSITY

❑ 목 적

본 프로젝트에서는 인터넷 강의의 회원 관리를 위해 회원 정보와 강의 수강 내역에 관한 정보를 처리하고 유지하는 정보 시스템을 구축하는 것을 목적으로 한다.

❑ 개 발 환 경

	OS	Visual Studio	Windows SDK	Language
버전	Windows 10 Pro	2017 Community	10.0.17134.0	C++

OS는 Windows 10 Pro를 사용하였다. Visual Studio는 무료 버전이며 쉽게 구할 수 있는 Visual Studio 2017 Community버전을 사용했으며, SDK 버전은 기본 값이 아닌 10.0.17134.0 버전을 사용하였다. 이 보다 높은 SDK버전을 사용하면 오류가 났으며, 구현에 사용한 언어는 C++이다.

❑ 프로젝트 기본요건

프로젝트의 기본 요건은 다음과 같다.

1. dat파일은 만들 txt파일 (listOfMember.txt, listOfLecture.txt, listOfPurchase.txt)은 무결성이 보장되어 있어야 한다.
2. 프로그램 실행 후 LecturePurchaseSystem을 처음 작동시킬 때만 txt파일을 통해 dat파일을 생성한다.

❑ main.cpp 설명

미리 제공된 헤더 파일 및 cpp파일과 이를 통합하여 main.cpp를 구현 하는 게 이번 프로젝트의 목표이다. 완성된 main.cpp는 아래 표와 같은 함수들로 구현되어 있다.

함수 이름	설명
printMenu	메인 메뉴를 보여주는 함수이다.
showMember	도큐먼트를 통해 제시된 showMember를 구현했다.
showLecture	도큐먼트를 통해 제시된 showLecture를 구현했다.
showPurchase	도큐먼트를 통해 제시된 showPurchase를 구현했다.
MemberTest	도큐먼트를 통해 제시된 MemberTest를 구현했다.
LectureTest	도큐먼트를 통해 제시된 LectureTest를 구현했다.
PurchaseTest	도큐먼트를 통해 제시된 PurchaseTest를 구현했다.
LecturePurchaseSystem	도큐먼트를 통해 제시된 LecturePurchaseSystem을 구현했다.
showSystemMenu	LecturePurchaseSystem의 메뉴를 보여주는 함수이다.
SeachingByID	도큐먼트를 통해 제시된 Search, Insert, Delete, Modify를 구현하였다. (CRUD 구현)
InsertingByID	
DeletingByID	
ModifyingByID	
DeleteMember	특정 아이디와 관련된 Member Record를 삭제하는 함수이다.
DeleteLecture	특정 아이디와 관련된 Lecture Record를 삭제하는 함수이다.
DeletePurchase	특정 아이디와 관련된 Purchase Record를 삭제하는 함수이다.

다음은 main.cpp를 구현하기 위해 필요한 .cpp 및 .h 파일에 대한 설명이다. 이 파일들은 프로젝트 당시 기본으로 주어져 있으며, 여기에 작성된 라이브러리를 이용하여 프로그램을 작성하여야 한다.

1. member.h

Member Class의 Field 자료구조 및 각 Field의 setter와 getter가 구현되어 있다. 또한 해당 객체의 길이를 반환해주도록 되어있다.

2. member.cpp

Member Class의 객체간의 비교/복사 연산에 사용될 오버로드된 연산자와 Pack/Unpack method 및 object의 출력과 입력에 사용될 method가 구현되어 있다.

3. lecture.h

Lecture Class의 Field 자료구조 및 각 Field의 setter와 getter가 구현되어 있다. 또한 해당 객체의 전체 길이를 반환해주는 함수도 구현되어 있다.

4. lecture.cpp

Lecture Class의 객체간의 비교/복사 연산에 사용될 오버로드된 연산자와 Pack/Unpack method 및 object의 출력과 입력에 사용될 method가 구현되어 있다.

5. purchase.h

Purchase Class의 Field 자료구조 및 각 Field의 setter와 getter가 구현되어 있다. 또한 해당 객체의 전체 길이를 반환해주는 함수도 구현되어 있다.

6. purchase.cpp

Purchase Class의 객체간의 비교/복사 연산에 사용될 오버로드된 연산자와 Pack/Unpack method 및 object의 출력과 입력에 사용될 method가 구현되어 있다.

❑ 프로그램 구성도 및 관계

프로그램 구성도를 그리기 전, 먼저 각 객체(Class)에 대한 정보를 알아야 한다. 구현한 객체는 총 3가지로 member, lecture, purchase가 있다.

1. member 객체

이 객체는 각각 한명의 회원에 대한 정보를 나타낸다. ID, Password, Name, Phone Number, Address, Mileage의 private field와 이 object의 여러 형식의 생성자, field들을 수정하거나 갱신할 method, object간 비교/복사 연산에 사용될 오버로드된 연산자로 구성된다. 클래스 Member에 대한 표는 다음과 같다.

Data	Type	Example
ID	variable string	ironman
Password	variable string	iron1234
Name	variable string	Tony Stark
Phone Number	variable string	010-1234-5678
Address	variable string	Seoul
Mileage	10 characters (Numeric)	0015407800

위의 method외에 Pack/Unpack method와 object 길이를 반환해 주는 method, object의 출력과 입력에 사용될 method로 구성된다.

2. lecture 객체

이 객체는 각각 하나의 Lecture에 대한 정보를 나타낸다. Lecture클래스는 Lecture 고유의 ID, Subject, Level, Price, 연장 여부, 수강일, 강사명, 교재명을 나타내는 데이터를 포

함하고, 여러 형식의 생성자, field들을 수정하거나 갱신할 method, object간 비교/복사 연산에 사용될 오버로드된 연산자로 구성된다. 클래스 Lecture에 대한 표는 다음과 같다.

Data	Type	Example
Lecture ID	12 characters (Numeric)	012345678900
Subject	variable string	Math
Level	1 characters	A/B/C/D 중 하나
Price	1 integer	300,000
Extension	1 characters	Y/N 중 하나
Due date	1 integer	30/60/90 중 하나
Name of teacher	variable string	Steve Rogers
Textbook	variable string	File Structure

위의 method 외에도 Pack/Unpack method와 오브젝트의 길이를 반환해주는 method 및 object의 출력과 입력에 사용될 method로 구성된다.

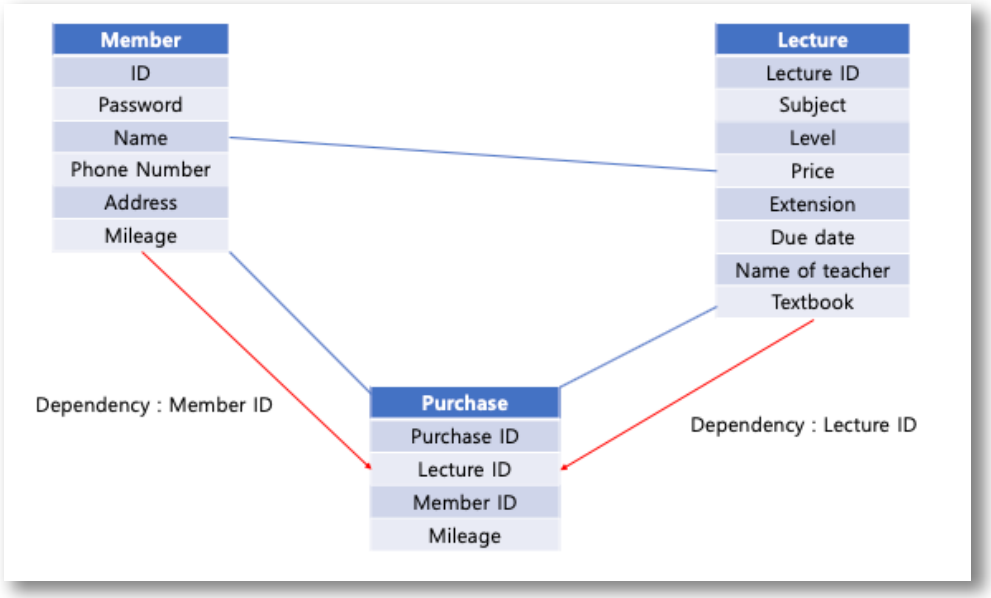
3. purchase 객체

이 객체는 각각 하나의 수강 내역에 대한 정보를 나타낸다. Purchase 클래스는 수강 내역 고유 ID, Lecture 고유 ID, 수강하는 회원의 ID, 추가 된 마일리지를 나타내는 데이터를 포함하고, 여러 형식의 생성자, field들을 수정하거나 갱신할 method, object간 비교/복사 연산에 사용될 오버로드된 연산자로 구성된다. 여기서 주의할 점은 Member ID나 Lecture ID는 실제 존재하는 데이터에 기반해야 한다. (무결성을 유지) 클래스 Purchase에 대한 표는 다음과 같다.

Data	Type	Example
Purchase ID	16 characters (Numeric)	0098765432101200
Lecture ID	12 characters (Numeric)	012345678900
Member ID	variable string	ironman
Mileage	10 characters (Numeric)	0000005000

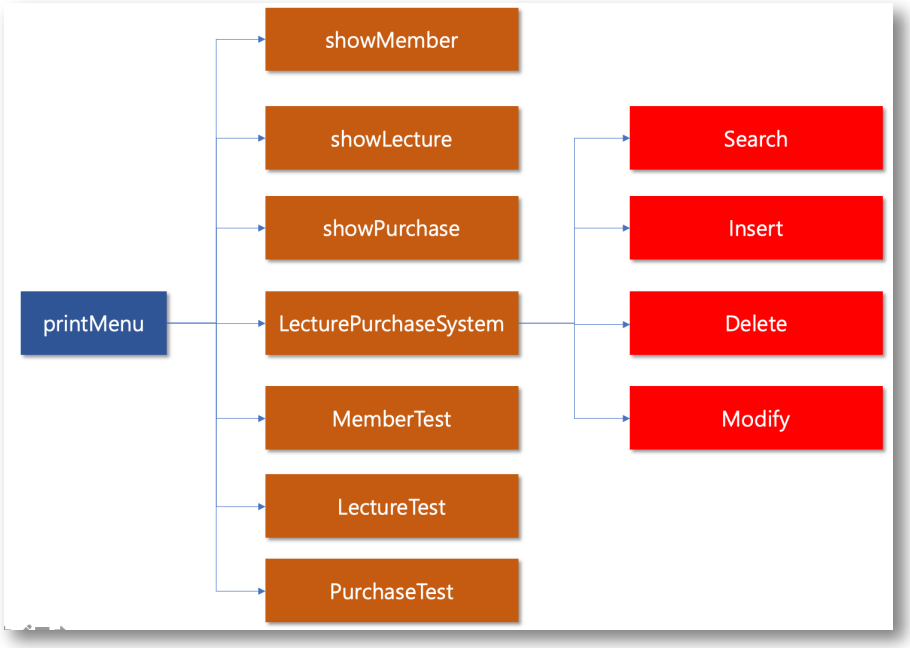
위의 method 외에 Pack/Unpack method와 오브젝트 길이를 반환하는 method, object 출력과 입력에 사용될 method로 구성된다. 다른 클래스와 달리 Purchase 클래스는 무결성을 유지하기 위하여 Lecture ID나 Member ID가 갱신되거나 삽입될 때, 기존의 Member 클래스나 Lecture 클래스에 해당 ID가 존재하여야 한다.

인터넷에는 무료이고 UML 다이아그램 (Reverse Diagram)을 그려주는 툴들이 존재한다. (Dia 등등) 하지만, 본 프로젝트의 구성은 간단하여, UML 다이아그램을 직접 그렸다.



클래스 간의 관계 (UML Diagram)

main.cpp에서 구현한 프로그램의 구성도는 다음과 같다.



main.cpp 함수(프로그램) 구성도

❑ 요구사항 기술

다음은 요구사항에 맞추어 main.cpp에 구현한 함수들의 설명이다.

1. showMember()

listOfMember.txt로부터 Member의 필드 값을 읽는다. 이 함수는 오버로딩 메소드를 확인하기 위한 Test Function이다.

2. showLecture()

listOfLecture.txt로부터 Lecture의 필드 값을 읽는다. 내용은 위와 같다.

3. showPurchase()

listOfPurchase.txt로부터 Lecture의 필드 값을 읽는다. 내용은 위와 같다.

4. memberTest()

Member 클래스에 추가한 Pack과 Unpack 함수를 확인하기 위한 Test Fuction이다. 먼저 listOfMember.txt로부터 데이터를 읽어들인다. 그 후 IOBuffer.h안에 있는 Pack(IOBuffer) 함수를 사용하여 fileOfMember.dat 파일을 작성한다. 작성된 fileOfMember.dat 파일에서 레코드를 출력하기 위해선 Unpack 함수를 사용한다.

5. lectureTest()

Lecture 클래스에 추가한 Pack과 Unpack 함수를 확인하기 위한 Test Function이다. 내용은 memberTest()와 동일하다.

6. purchaseTest()

Purchase 클래스에 추가한 Pack과 Unpack 함수를 확인하기 위한 Test Function이다. 내용은 memberTest(), purchaseTest()와 동일하다.

7. lecturePurchaseSystem()

생성된 각 데이터파일(fileOfMember.dat, fileOfLecture.dat, fileOfPurchase.dat)을 통해서 검색, 삽입, 삭제, 수정을 지원하는 대화식 프로그램이다. Search/Insert/Delete/Modify의 과정들은(CRUD) 각각의 고유 ID들을 통해 이루어진다.(Unique 해야한다.) 모든 경우에서 참조 무결성을 유지해야하며, 수정에서는 ID를 제외한 모든 필드가 가능하다.

8. Search (Select 성격)

사실 데이터베이스의 핵심은 Search에 있다고 생각한다. 얼마나 빨리 필요한 정보를 찾아내는지가 서비스에 중요한 요소로 작용하기 때문이다. 본 프로젝트에서 Search에 데이터베이스와 파일 시스템에서 널리 사용되는 트리 자료구조의 일종으로 B-트리(B-tree)를 구현해 보고 싶었다. 또한 indexing 작업인 Key를 구현해 보고 싶었지만, 구현방법이 너무 어려워, 구현하지는 못하였다.

요번 Search는 mysql의 full-search와 같은 방식으로 구현하였다. (원하는 값을 찾을 때까지 서치하는 방식) Record의 ID로 검색 한 후 레코드를 출력하게끔 하였으며 MemberID나 Lecture ID도 검색 지원이 되도록 만들었다. 또한 함수를 호출할 때, 플래그 값을 두어 재사용성이 용이하도록 만들었다.

9. Insert

본 프로젝트에서 구현해야 하는 ID들은 전부 고유 ID이다. 즉, mysql의 unique key와 같은 성격의 ID를 구현해야 한다. 하지만, key(indexing)을 구현하지 못하였으므로, 마찬가지로 full-search로 ID를 순차적으로 검색 후, 중복되는 값이 없을 때, 해당 ID에 맞는 데이터들을 삽입할 수 있도록 하였다.

10. Delete

무결성을 유지하기 위하여 record ID를 사용하여 검색 후, 무결성을 유지하며 해당 ID에 대한 record 들이 삭제되게끔 구현하였다.

11. Modify (Update 성격)

record의 ID를 사용하여 검색 후, 존재하는 ID라면 해당 record의 대한 정보를 수정할 수 있도록 구현하였다. modify과정에서 형식에 맞지 않는 data를 입력이 되면 에러 메세지를 출력하도록 하였다.

❑ 연 습 문 제

21 번.

가변길이 레코드를 처리할 때, 삭제된 공간만큼 Shift하는 것으로 처리하였다. 따라서 삭제된 레코드를 인식하는 일이 없었다.

22 번.

레코드가 삭제되면 Shift를 하여 빈 공간을 채우는 식으로 구현하였다. 또한 새로운 레코드가 삽입되면 맨 뒤에 삽입하게끔 구현하였다.

23 번.

가변길이 레코드를 처리할 때, 삭제된 레코드 앞에 특수문자와 같은 Special Character를 추가하여 처리하면, 매번 해당 레코드의 전체 길이를 구하여 처리해야 한다. 또한 이는 메모리를 잡아먹는 리소스 낭비나 다름이 없다. 특히 현재 Record들은 특정 B-tree에 의해 정렬된 것이 아니기 때문에 Search, Update, Delete에는 전부 full search가 이루어진다.(순차 탐색) 따라서 여기서 Shifting하여 빈 공간을 아예 없애어 full search의 시간을 줄이는 방식으로 처리하여도 괜찮다고 판단된다. 추후 Indexing을 구현하게 되면 Insert, Search, Update, Delete 전부 수정해야 한다. (Insert에는 고유 ID의 값을 입력하게 될 때, 순차 탐색이 이루어진다.)

24 번.

가변길이 레코드의 Delete를 Shifting을 통해 구현하였기 때문에 현재는 Append를 수정할 필요가 없지만, 만약 Record에 특정 Index가 추가되면 Search, Update, Delete, Insert가 전부 수정되어야 한다. 이에 따라 Append 연산도 당연히 수정되어야 한다.

25 번.

update 연산을 추가하였다. 현재는 Shifting방식을 통해 구현하였기 때문에 variable length string을 입력할 때는 문제가 없고 또한 원래의 string보다 작은 크기의 string이 입력되어야 한다 하더라도 Data Compaction에 대해서는 문제가 발생하지 않는다. 또한, 정해진 형식이 있는 데이터에 잘못된 값이 입력이 되었을 경우, 에러 핸들링이 되도록 하여, 수정이 불가하도록 만들었다.