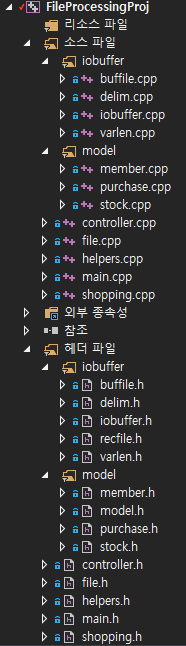
**File Processing Project #1**

*By 20141500 권태국*

# **프로젝트 자료구조와 파일에 대한 설명**



나의 프로젝트 소스 파일 구조는 위와 같다.

1. Iobuffer

IOBUFFER와 관련된 것들이 위치하고 있다.

1. Model

Member, Stock, Purchase와 같은 Data Model들이 위치하고 있다.

1. Controller

데이터 모델에 대한 검색, 삭제, 삽입과 같은 각종 control관련 된 기능들이 위치하고 있다.

1. File

파일과 관련된 것들이 위치하고 있다. 리스트파일을 읽고 자동으로 생성하는 기능, 데이터파일을 읽고 쓰는 기능들이 있다.

1. Helpers

프로젝트 전반적으로 필요한 기능들이 위치하고 있다. 예를 들면, 숫자와 문자열 간의 변환 같은 것이다.

1. Shopping

Online Shopping System의 UI와 기능들이 위치하고 있다.

1. Main

위의 모든 기능들을 사용하는 main module이다.

나는 data들을 메모리상에 저장하는 데에 있어 C++의 container중 하나인 vector를 사용하였다. 그리고 member, stock, purchase각각이 dat파일 상의 어느 recaddr에 저장 되어있는 지를 객체안에 저장하였다.

# **프로젝트 요구사항 완료 보고**

## Basic class

프로젝트의 기본적인 클래스 Member, Stock, Purchase를 작성하였다.

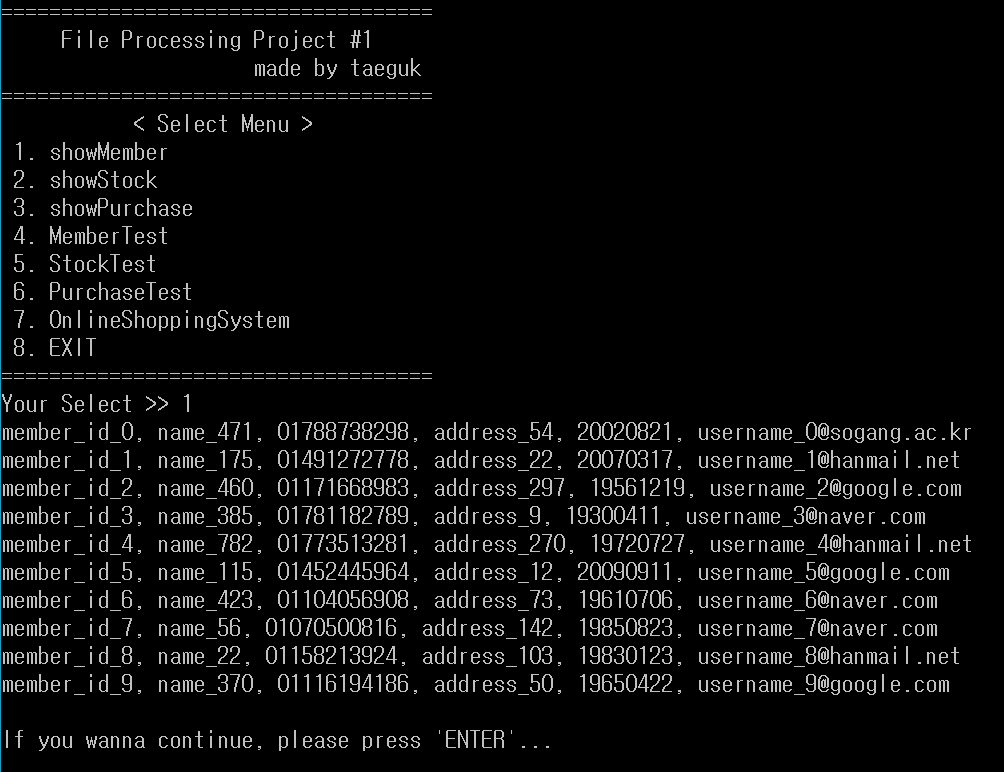
나는 프로젝트의 구조에서 이 3개의 클래스를 data model이라는 용어로 지칭하였다.

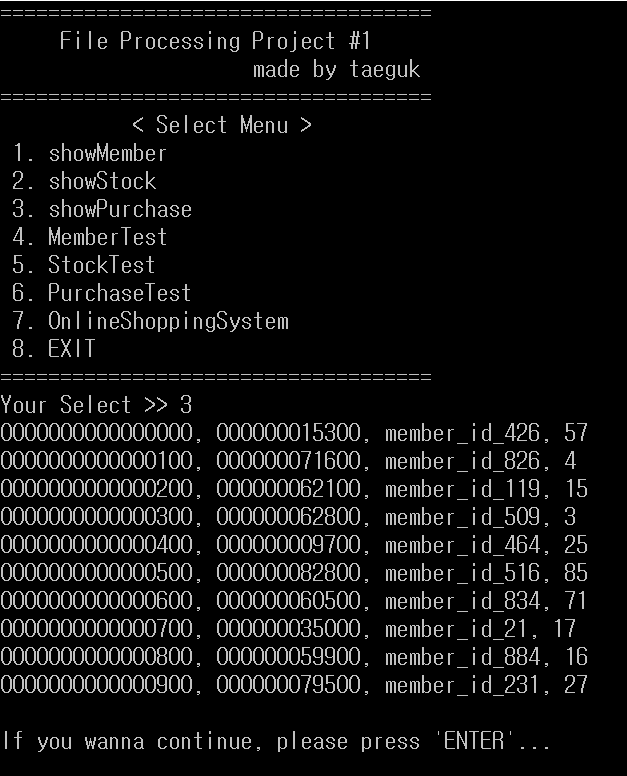
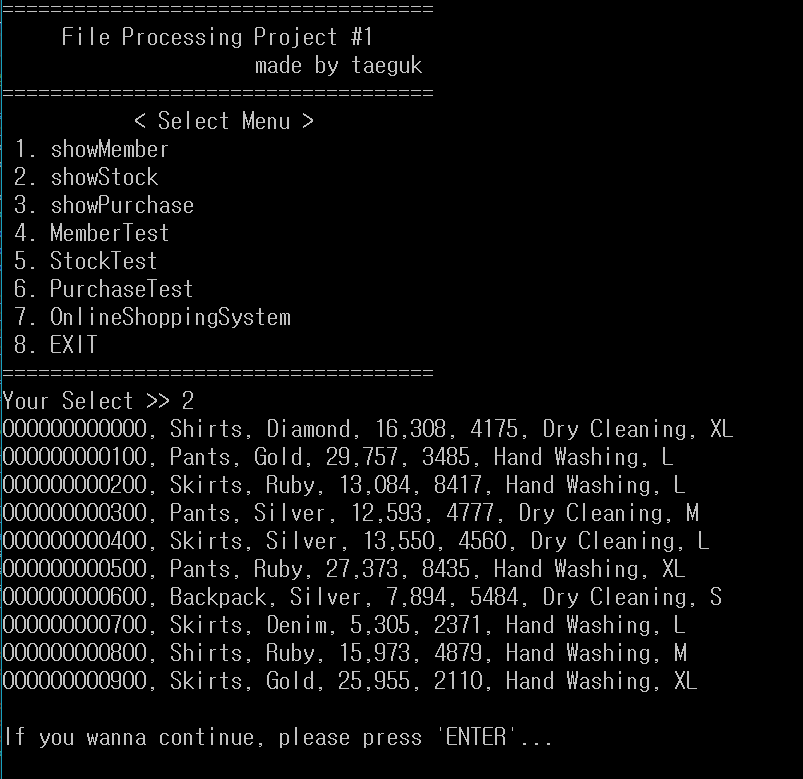
문서에서 요구한 대로 잘 구현하였다.

실제 회원/상품/구매 목록을 작성하는 것에 대해서는 직접 프로그램을 작성하여 생성하였다. 랜덤하게 값들을 선정하였고, Purchase관련해서는 Member와 Stock에 대한 무결성도 고려하여 값들을 선정하였다.

## Adding methods to basic classes

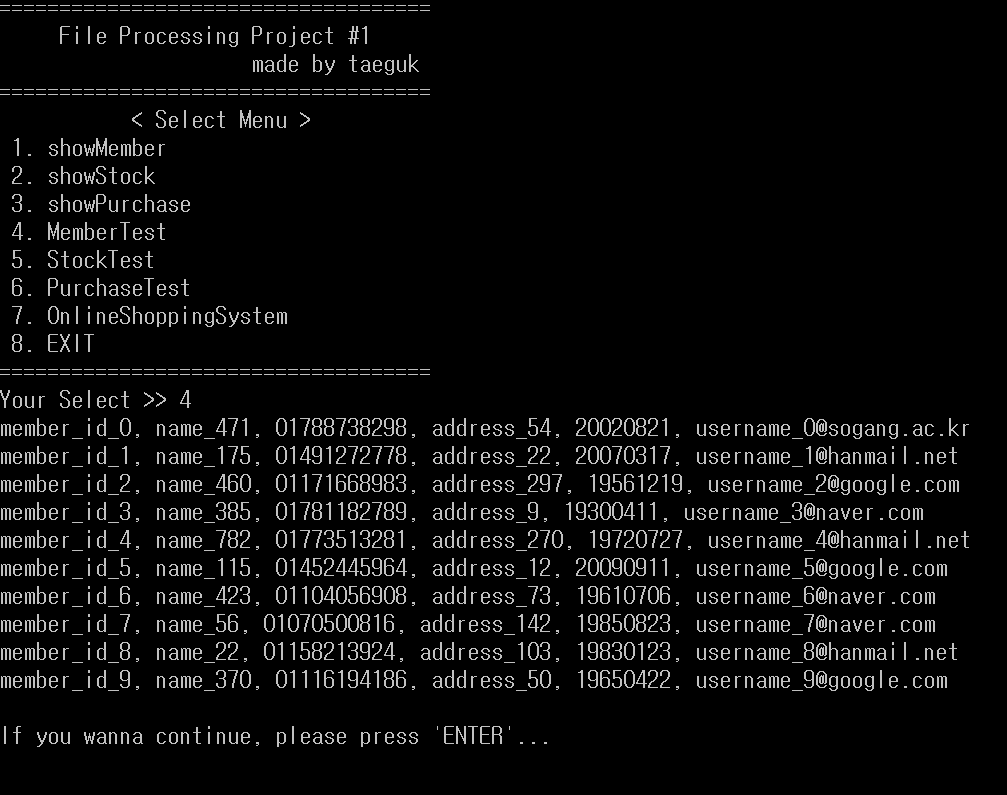
문서에서 요구하는 데로 구현하였다.

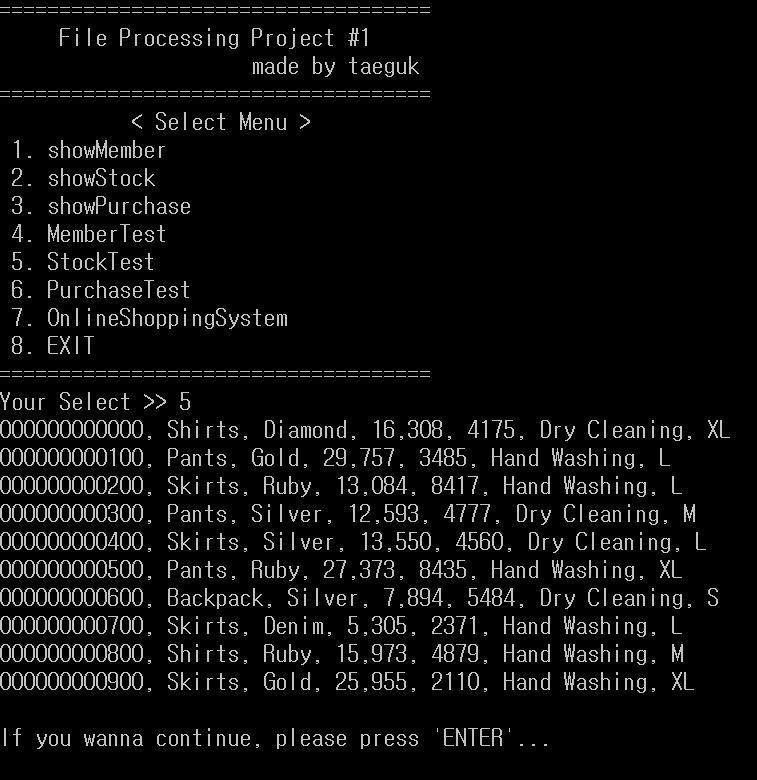


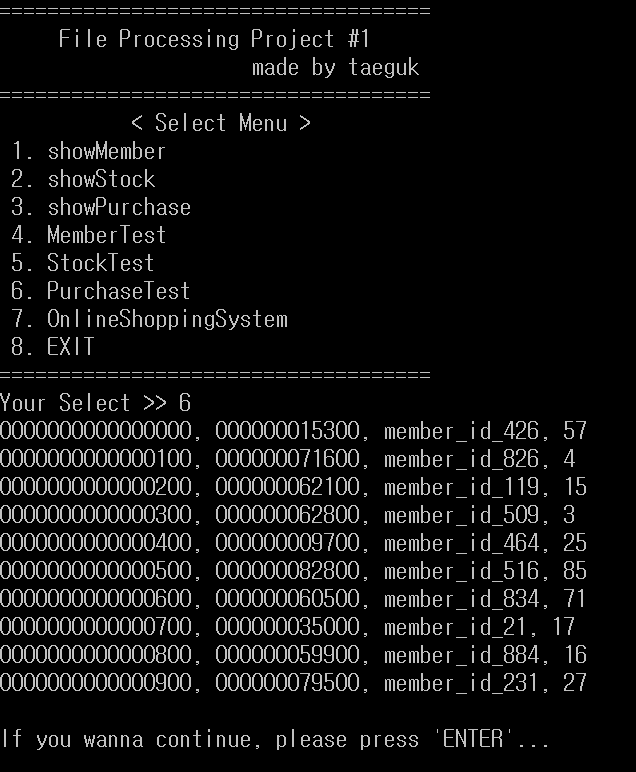


위는 요구사항에서 요구한 showMember, showStock, showPurchase 에 대한 구현 결과이다. 보면 요구사항에 알맞게 잘 작동되는 것을 알 수 있다.

## Using IOBuffer







위는 요구사항에 알맞게 MemberTest, StockTest, PurchaseTest를 구현한 것을 실행한 결과이다.

## Deleting and Updating Records

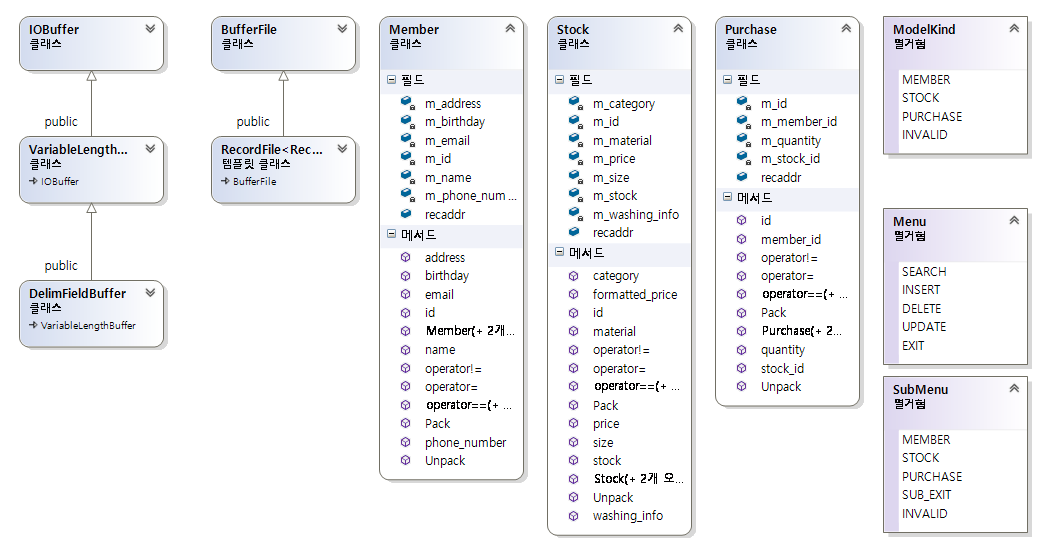
나는 요구사항에 맞게 구현하였다. 핵심 포인트들을 나열해보도록 하겠다.

나는 첫째, 검색/삽입/삭제/수정을 다 지원하는 대화식 프로그램을 작성하였다. 수정은 키를 제외한 모든 필드가 가능하며, 모든 경우에서 참조 무결성을 유지하였다.

예를들어, Stock 레코드가 삭제되면 Stock ID를 외래키로 가지고 있는 Purchase들도 자동으로 삭제된다.

그리고 삽입과 삭제, 그리고 수정을 IOBUFFER를 활용해 dat파일에 바로 작업하였다. 삭제 같은 경우 특정 레코드를 특수기호로 표시하여 삭제된 것을 표시하였다. Dat 파일 READ시 이렇게 삭제 표시된 레코드들을 생략하도록 구현하였다. 이렇게 DAT파일에 바로 특수기호를 이용하여 삭제를 구현하였기 때문에 효율적이다.

# **클래스 다이어그램**



클래스 다이어그램은 위와 같다.