User-friendly Time table program
: Class Mate

TEAM 9
200911385 박기남
200911425 조서경
200911426 조성완
200911427 조아라

Index

Ⅰ. Introduction

1) OOAD & UML

① 소프트웨어를 개발하는 하나의 방법론으로 모든 소프트웨어 시스템의 주요 기본요소를 사물을 가리키는 객체와 그 객체들을 하나의 집합으로 묶은 클래스로 구성하는 객체지향적인 분석과 설계 방법을 말한다. 객체지향적 이란 것은 현실세계에 실제 존재하는 사물, 즉 객체들을 지향한다는 것이다. 그리함으로써 보다 현실세계의 개념을 편리하고 빠르게, 즉 실용적으로 이용할 수 있다.

UML(Unified Modeling Language)은 기존의 Booch 방법론, Rumbaugh 등의 OMT(Object Modeling Technique), 그리고 Jacobson의 OOSE 방법론 등을 연합하여 만든 모델링을 위한 표준으로 객체지향 시스템 모델을 작성하기 위한 객체지향적 분석과 설계 개념의 표기법을 제공한다.

그러므로 UML 자체가 방법론이라 할 수는 없으나 방법론을 수행하기 위한 최적의 도구라 할 수 있다.
객체지향 개발 방법론의 특징 중 하나가 전통적 개발 방법론보다 분석, 설계에 할애하는 시간과 노력이 많다는 것은 기술된 바 있다. 그러므로 분석 설계 단계를 표준화 된 다이어그램과 모델링 툴을 이용해 수행하고 그 산출물이 객체지향 언어를 통해 체계적으로 번역되어 소스 코드가 생성될 수 있게 하는 것이 매우 중요하며 이것이 프로젝트 성공의 관건이라 할 수 있다.

UML은 Class Diagram, Object Diagram, Use Case Diagram 등 여러 Diagram을 기반으로 객체지향 어플리케이션을 개발하기 위한 분석 및 설계를 뒷받침하는 훌륭한 방식을 제공하고 있다.

② 본 Project는 "starUML" TOOL을 이용하여 Diagram을 제작하여 설계수행을 하였다.

Ⅱ. Plan

1) Define Draft Plan & Investigation Report

① A draft project plan

ⅰ. 시간표출력 및 관리와 더불어, 출석, 성적관리, 시간표의 학기별 보관, 과목별 일정 등을 관리할 수 있는 시간표 프로그램이다. 이는 기존 프로그램이 만족시키지 못했던 가시성, 실용성, 편의성 등을 십분 개선하여 보다 효율적인 프로그램을 만드는데 에 초점을 맞추었다.

ⅱ. Schedule

11 / 3	Plan(Draft plan)
11 / 4	Plan(Prototyping)
11 / 5	Analysis (Make use-case description & diagram)
11 / 9	Analysis (Make diagrams & operation steps)
11 / 17	Design & Bulid
11 / 29	Implementation
12 / 8	Final Demo

② Software Requirements Specification

ⅰ. System Interface

- Functional Requirements

· 시간표 제작, 관리
· 과목별 성적 관리
· 과목별 일정 관리
· 과목별 출석 관리
· 암호 방식 도입

- Non-Functional Requirements

· 암호입력을 통한 개인정보 보호
· 적은 용량에 의한 PC간 이동의 편의성
· 365일 24시간 사용이 가능
· Data Base 관리 가능

ii. User Interface

- Not to do list
  · Data Base 삭제
  · Invalid Input
  · Using in LINUX (recommended using in Window)

- To do list
  · File I/O 를 통하여 정보 저장 및 출력이 가능
  · 실제 날짜 및 시간에 맞추어 작성
  · Command창에서 일정이 저장되어 있는 Data Base에 변경사항 적용

iii. Hardware Interface

- 모든 입력은 키보드장치를 통하여 입력
- Window OS 기반
- 최소 10MB의 RAM 요구
- 32bit Computer 기반

iv. Software Interface

- Name : ClassMate
- Specification number : 1.0
- Version number : 1.0

v. Risk & Solution

- 암호입력 없이 DB에 접근
  => DB파일을 숨김 파일로 지정. 외부에서 접근 불가케 함

2) Term Dictionary

① Data base : 본 프로그램에서 지속적 관리가 필요한 정보를 ".txt" 의 확장자로 저장한 텍스트 파일

② OS : 운영체제

③ 32bit Computer : int 변수를 4byte로 갖는 형식의 Computer

3) Draft System Architecture

① Classes & Methods

ⅰ. Class Identify

```
Class Identify
{
        Input_Password()
        {}

        Check_Password()
        {}

        Change_Password()
        {}
}
```

ⅱ. Class FileIO

```
class FileIO : public Identify
{
        InFile_Open()
        {}

        InFile_Close()
        {}

        OutFile_Open()
        {}

        OutFile_Close()
        {}

        Set_FileName()
        {}

        Get_FileName()
        {}
}
```

iii. Class SubManager

```
Class SubManager : public FileIO
{
        Delete_Subject()
        {}

        Change_SubInfo()
        {}
}
```

iv. Class Subject

```
Class Subject : public FileIO
{
        Add_Subject()
        {}

        Show_List()
        {}
}
```

v. Class Grade

```
Class Grade : public FileIO
{
        Input_Grade()
        {}

        Read_FGrade()
        {}

        Cal_Grade()
        {}

        Show_Grade()
        {}
}
```

vi. Class Date

```
Class Date : public FileIO
{
        Input_StartDate()
        {}

        Set_Subject()
        {}

        Input_Date()
        {}

        Show_Schedule()
        {}

        Show_Menu()
        {}

        Add_Schedule()
        {}

        Add_ASubject()
        {}

        Del_ASubject()
        {}

        Extend_ASubject()
        {}

        Check_Attention()
        {}
}
```

Ⅲ. Analysis

1) Use-Case

① Description

ⅰ. Use Case 명 : 사용자 인증

1. 개요

사용자가 패스워드 입력을 통해 자기 자신을 인증한다.
('Class Mate'는 여러 유저가 사용할 수 있는 시스템이므로 각 사용자마다 이름과 패스워드를 입력받아서 인증하는 과정이 필요하다.)

2. Flows of Events

사용자(액터)	Class Mate(시스템)
	패스워드를 액터에게 묻는다.
-> 패스워드를 입력한다.	
	->패스워드가 일치하는지 확인한다.
-> 패스워드가 일치하면 시스템에 로그인된다.	

ⅱ. Use Case 명 : 수강과목추가

1. 개요

사용자가 자신이 수강하는 과목을 추가하고 그 과목의 정보를 입력한다.

2. Flows of Events

사용자(액터)	Class Mate(시스템)
사용자가 수강과목 추가 메뉴를 선택한다.	
	->액터에게 과목 정보의 입력을 요청한다. (과목명, 교수명, 강의시간, 강의실, 학점)
-> 과목 정보를 입력한다.	
	-> 사용자의 과목정보 저장용 데이터베이스에 과목 정보를 저장한다.

iii. Use Case 명 : 수강과목삭제

1. 개요

사용자가 자신이 추가했던 과목을 삭제한다.

2. Flows of Events

사용자(액터)	Class Mate(시스템)
사용자가 수강과목삭제 메뉴를 선택한다.	-> 과목정보 저장용 데이터베이스에서 해당과목에 관한 모든 것을 삭제한다.

iv. Use Case 명 : 수강과목조회

1. 개요

사용자가 자신이 수강하고 있는 과목의 리스트를 조회한다.

2. Flows of Events

사용자(액터)	Class Mate(시스템)
사용자가 수강과목조회 메뉴를 선택한다.	-> 과목정보 저장용 데이터베이스에서 이 사용자가 수강하고 있는 과목의 정보를 읽고 콘솔에 출력한다.

v. Use Case 명: 성적관리

1. 개요

사용자가 자신이 수강하고 있는 과목의 성적을 관리할 수 있다.

2. Flows of Events

사용자(액터)	Class Mate(시스템)
사용자가 성적관리 메뉴를 선택한다. -> 사용자가 과목별로 성적을 입력한다.	
	-> 사용자에게 입력받은 성적과 총 수강학점수를 이용하여 평점을 계산한다.

vi. Use Case 명 : 일정관리

1. 개요

사용자가 자신이 수강하고 있는 과목에 관한 일정을 관리한다.

2. Flows of Events

2.1 Main Flow

사용자(액터)	Class Mate(시스템)
사용자가 일정관리 메뉴를 선택한다. -> 사용자가 조회하고 싶은 날짜를 선택한다.	
	-> 시스템이 사용자가 선택한 날짜의 일정관리용 데이터베이스를 조회한다. (과목검색)
-> 추가 기능을 선택할 수 있다.	
	-> 사용자가 선택한 기능을 수행한다.

2.2 Alternative Flows

A-1. 사용자가 휴강처리를 선택할 경우

시스템이 사용자가 선택한 날짜의 일정관리용 DB에서 해당과목의 정보를 삭제한다.

A-2. 사용자가 연강처리를 선택할 경우

시스템이 사용자가 선택한 날짜의 일정관리용 DB에서 해당과목의 강의시간을 연장한다.

A-3. 사용자가 보강처리를 선택할 경우

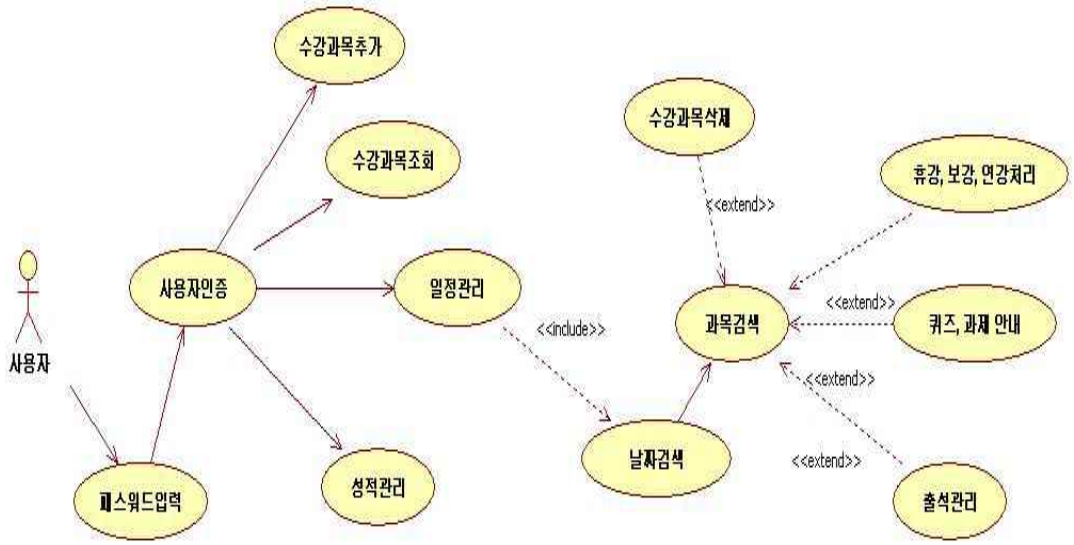
시스템이 사용자가 선택한 날짜의 일정관리용 DB에 해당과목의 보강정보를 저장한다.

A-4. 사용자가 출석관리를 선택할 경우

사용자가 날짜를 선택하고, 과목을 선택한 후 출석여부를 입력한다.

-> 시스템이 사용자가 선택한 날짜의 일정관리용 DB에 해당과목의 출석여부를 저장한다.

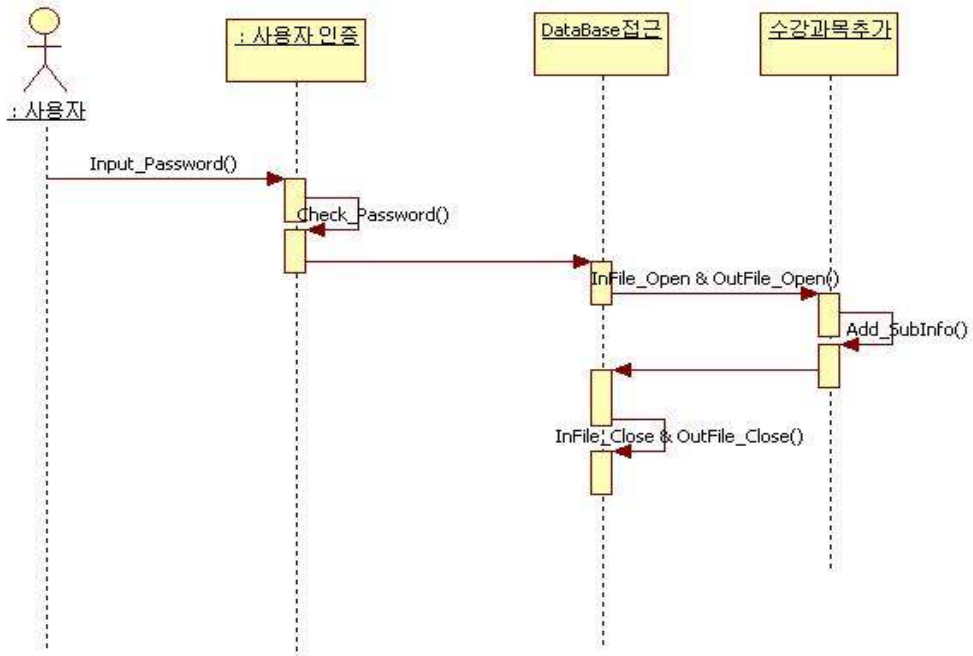
② Diagram


수강과목추가
수강과목조회
수강과목삭제
휴강, 보강, 연강처리
<<extend>>
사용자인증
일정관리
<<include>>
과목검색
<<extend>>
퀴즈, 과제 안내
사용자
<<extend>>
패스워드입력
성적관리
날짜검색
<<extend>>
출석관리

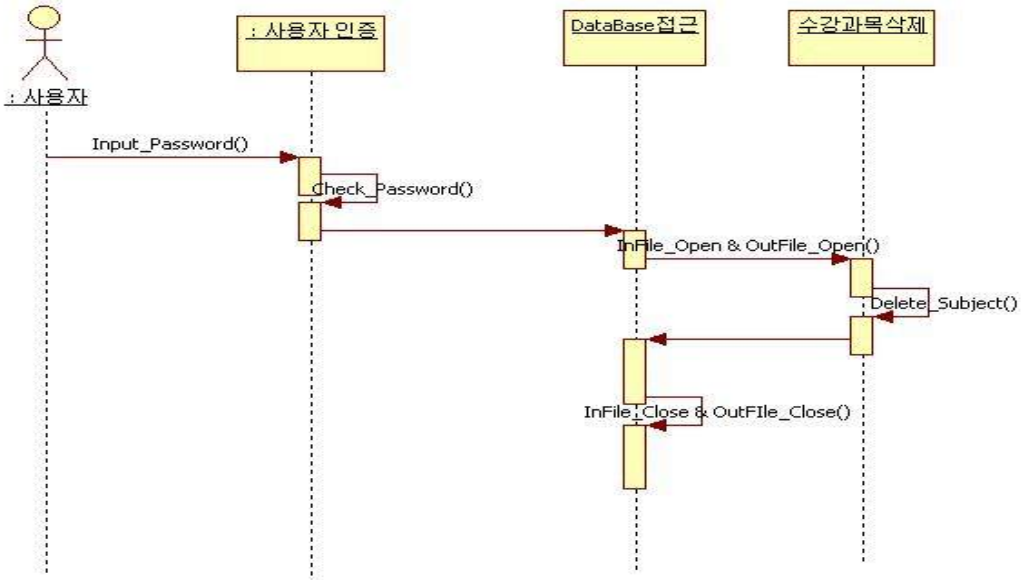
2) Sequence

① Diagrams

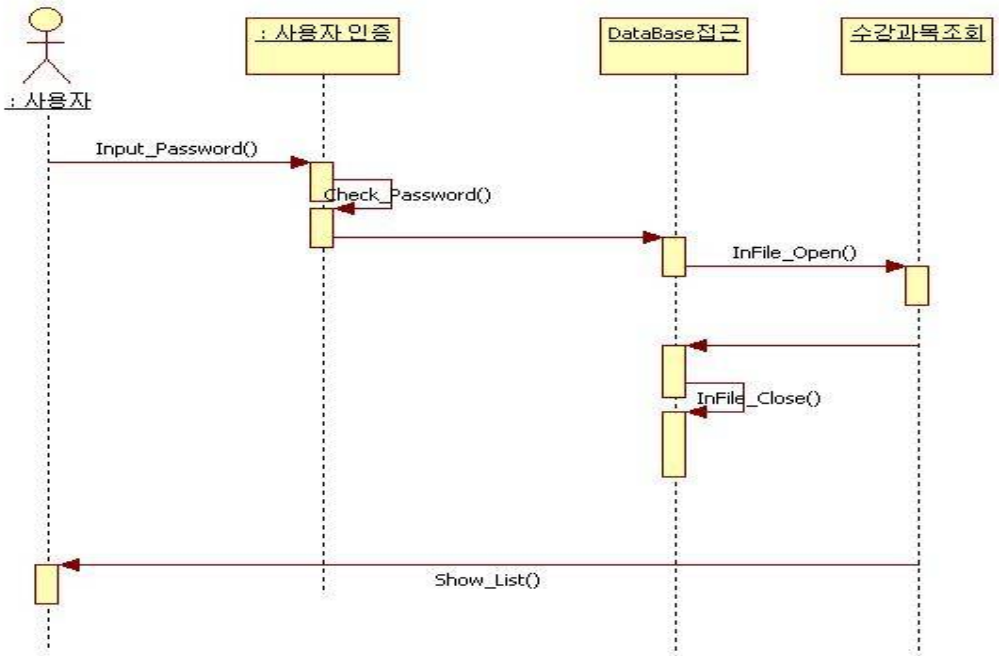
ⅰ. 수강과목추가



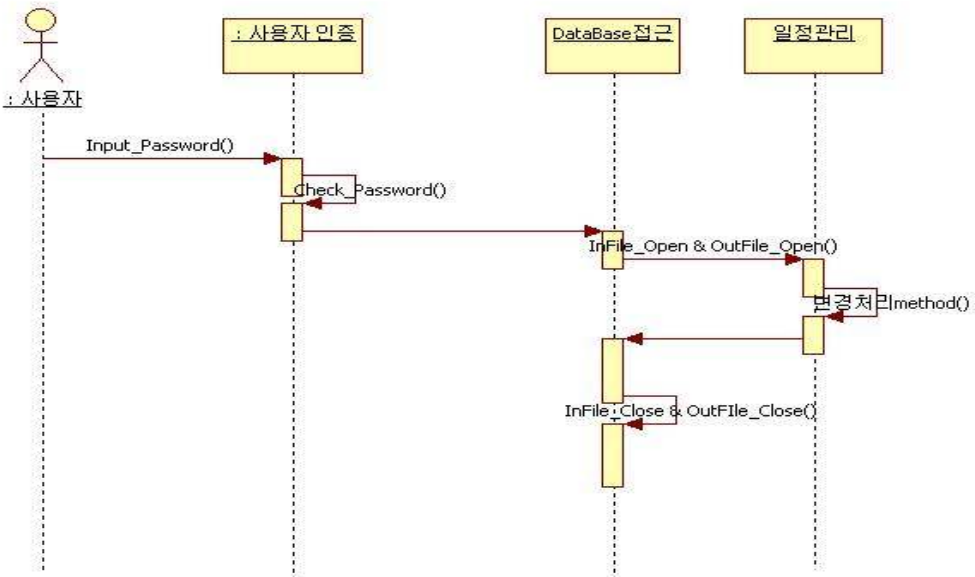
ⅱ. 수강과목삭제



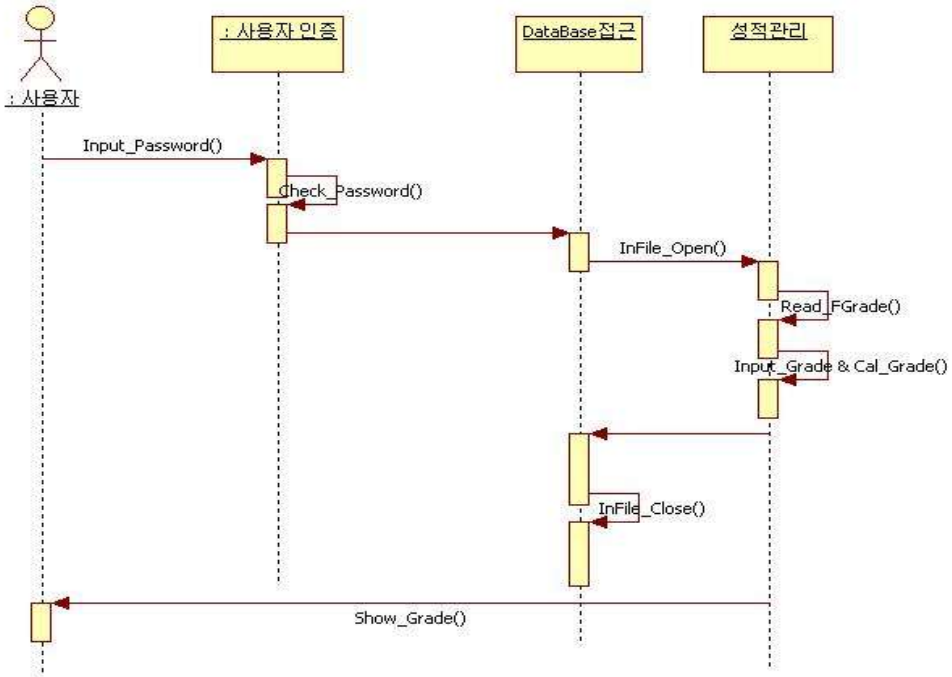
iii. 수강과목조회



iv. 일정관리

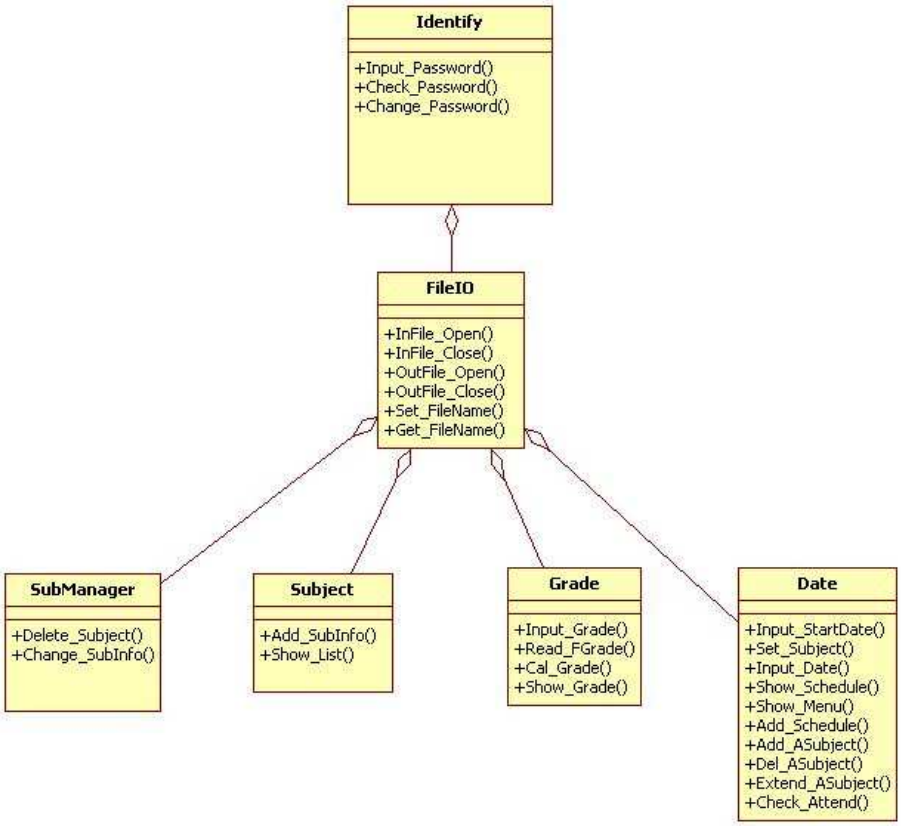


v. 성적관리


: 사용자
: 사용자 인증
DataBase접근
성적관리
Input_Password()
check_Password()
InFile_Open()
Read_FGrade()
Input_Grade & Cal_Grade()
InFile_Close()
Show_Grade()

3) Class

① Diagrams

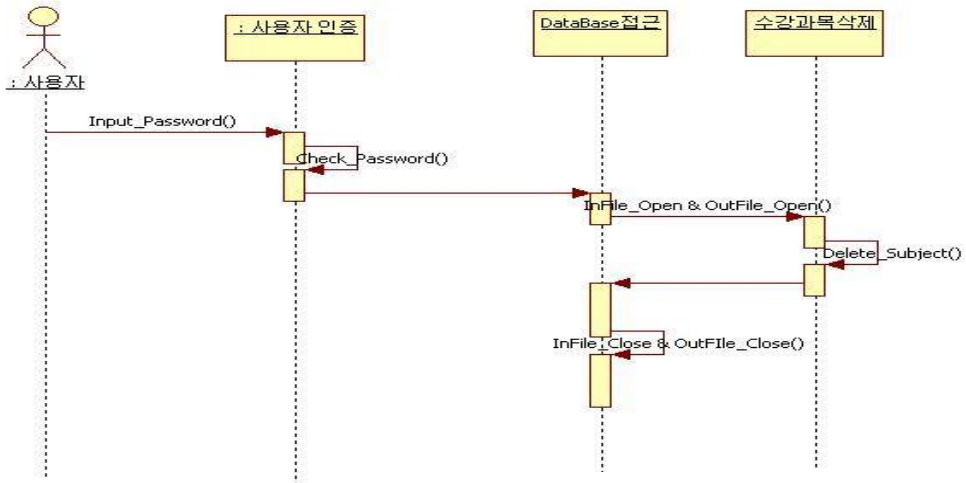


4) Analyze Steps & Glossary(Contracts)

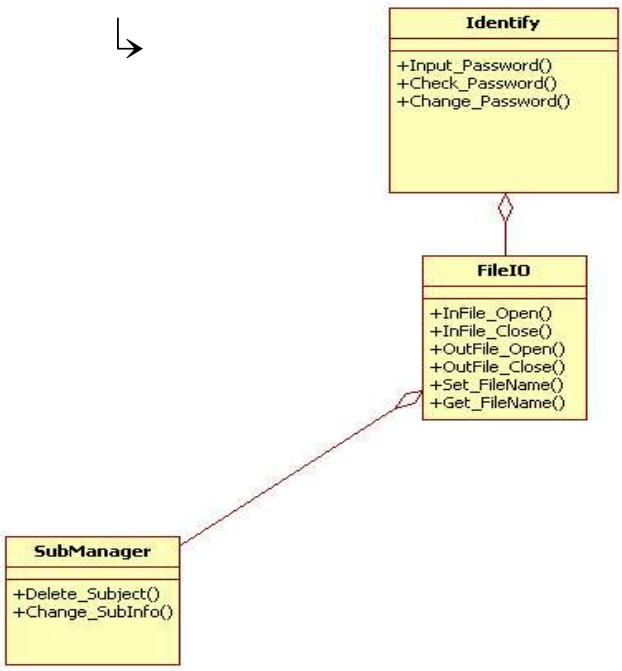
① Analyze Steps

ⅰ. 수강과목삭제

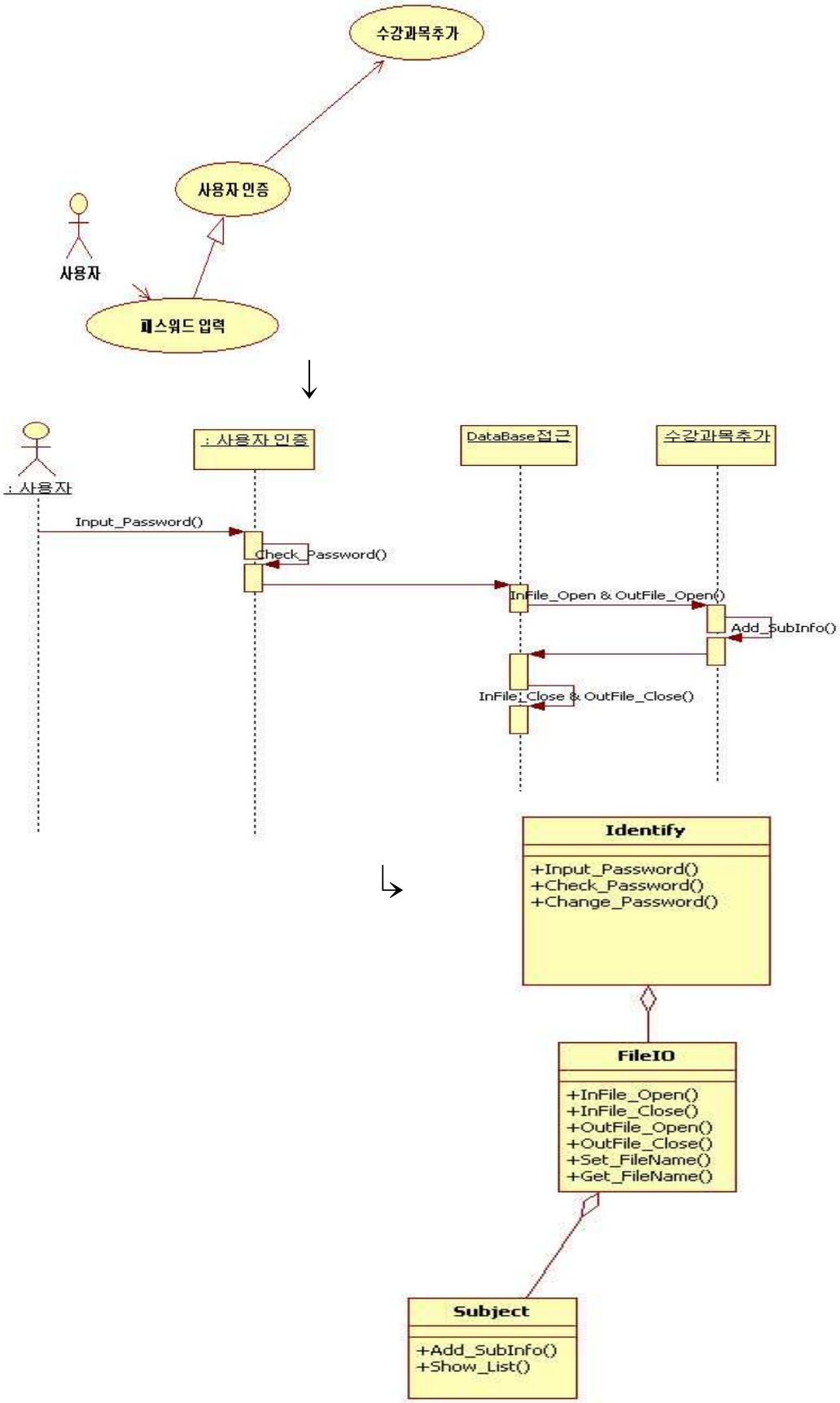
↓



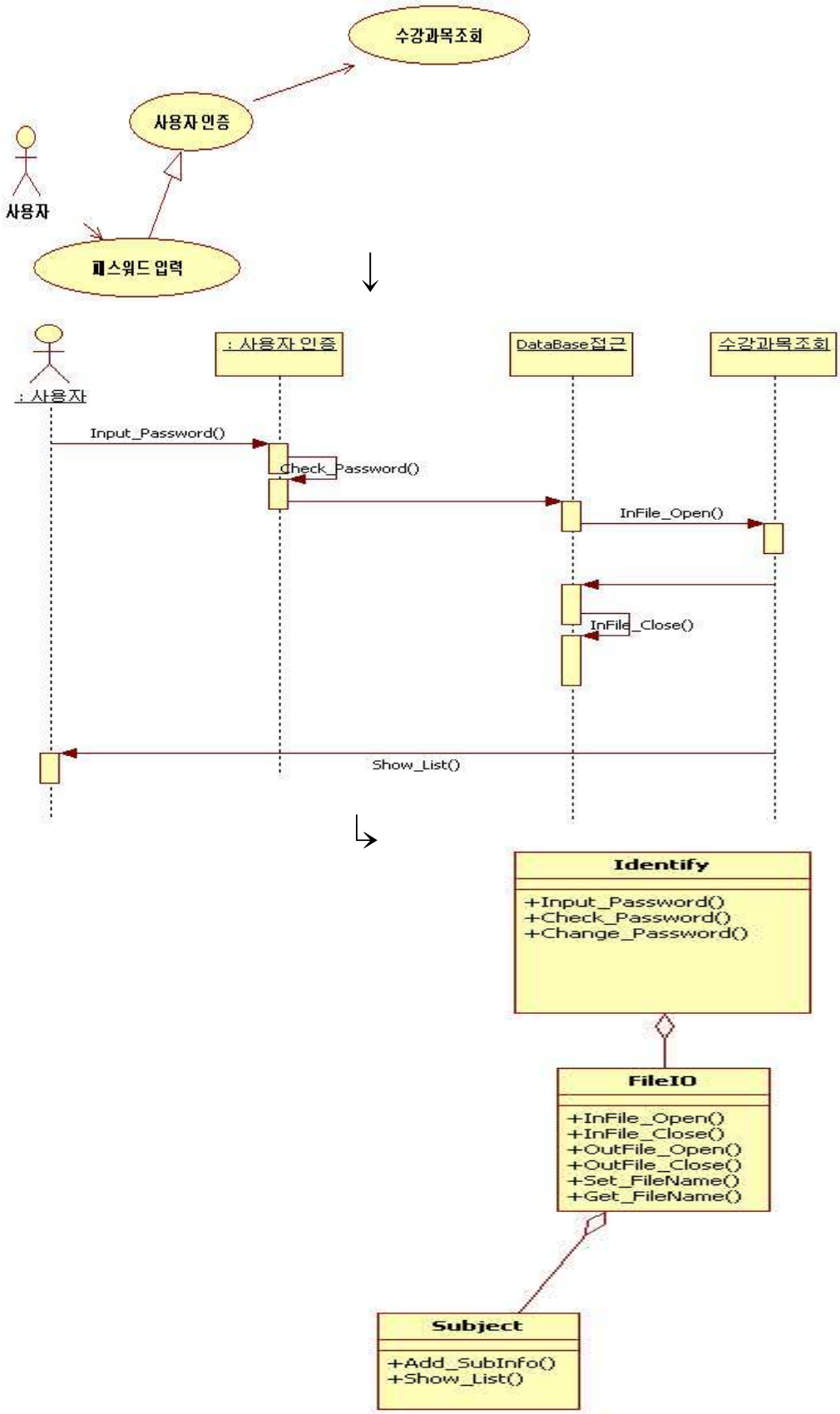
↳



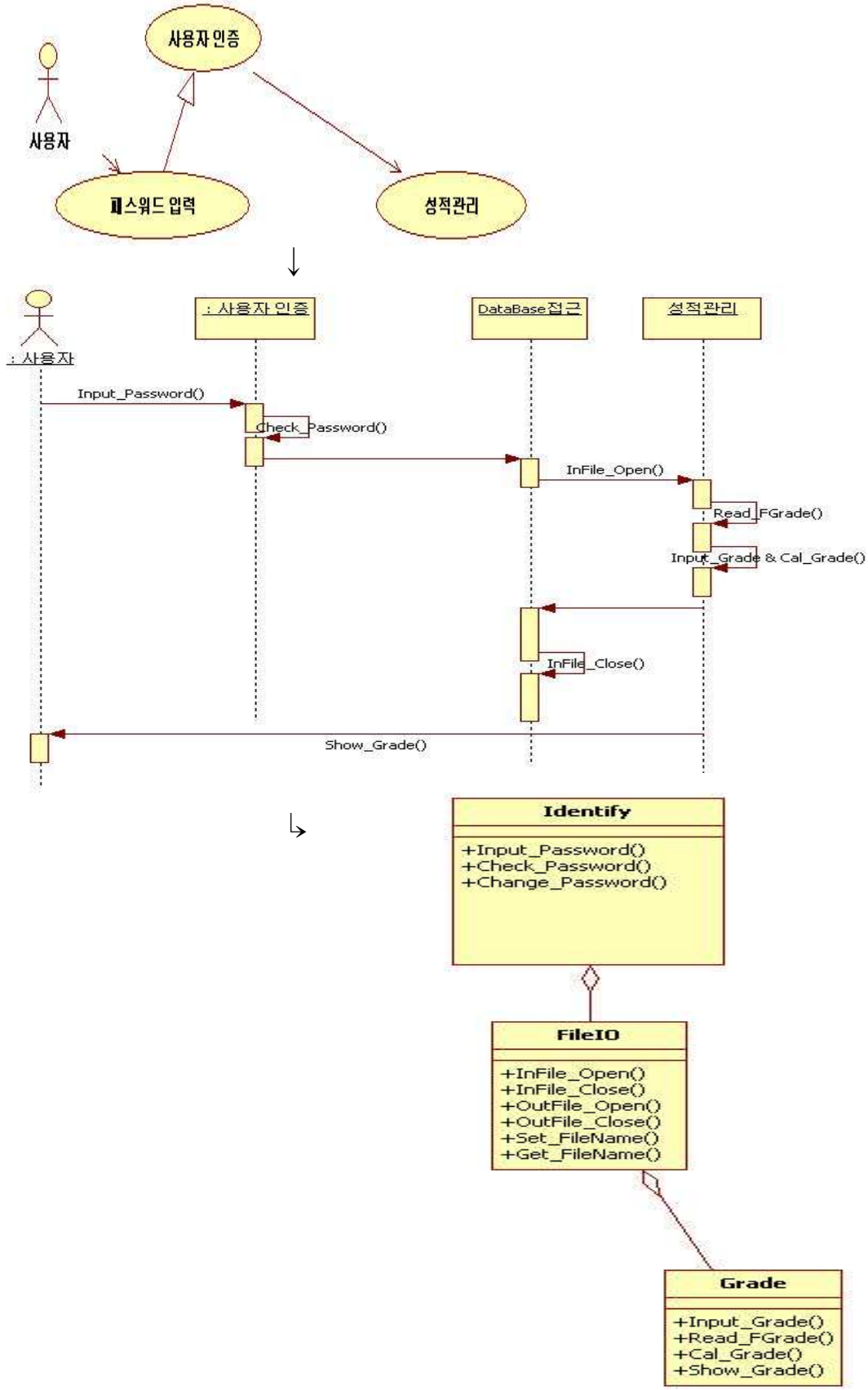
ii. 수강과목추가


수강과목추가
사용자 인증
사용자
패스워드 입력
: 사용자
: 사용자 인증
DataBase접근
수강과목추가
Input_Password()
Check_Password()
InFile_Open & OutFile_Open()
Add_SubInfo()
InFile_Close & OutFile_Close()
Identify
+Input_Password()
+Check_Password()
+Change_Password()
FileIO
+InFile_Open()
+InFile_Close()
+OutFile_Open()
+OutFile_Close()
+Set_FileName()
+Get_FileName()
Subject
+Add_SubInfo()
+Show_List()

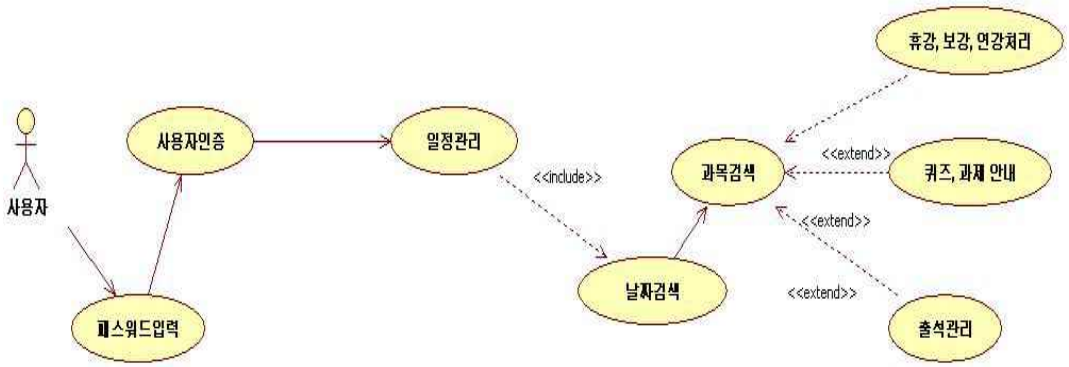
iii. 수강과목조회

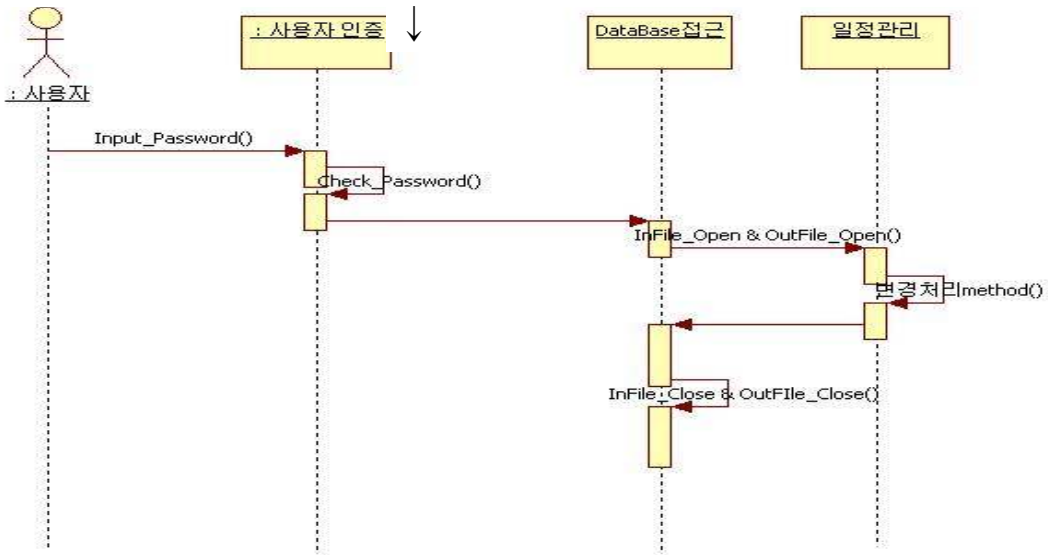

수강과목조회
사용자 인증
사용자
패스워드 입력
: 사용자
: 사용자 인증
DataBase접근
수강과목조회
Input_Password()
Check_Password()
InFile_Open()
InFile_Close()
Show_List()
Identify
+Input_Password()
+Check_Password()
+Change_Password()
FileIO
+InFile_Open()
+InFile_Close()
+OutFile_Open()
+OutFile_Close()
+Set_FileName()
+Get_FileName()
Subject
+Add_SubInfo()
+Show_List()

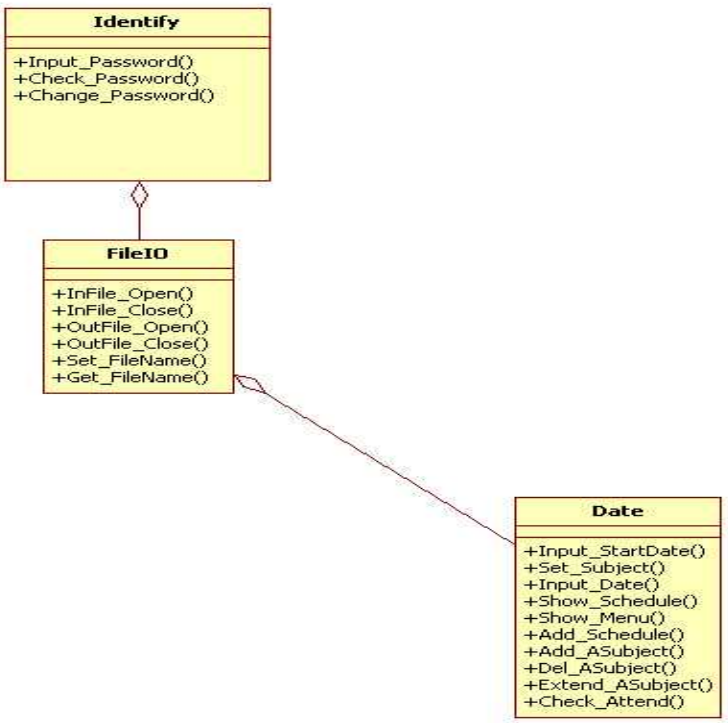
iv. 성적관리


사용자인증
사용자
패스워드입력
성적관리
: 사용자
: 사용자 인증
DataBase접근
성적관리
Input_Password()
Check_Password()
InFile_Open()
Read_FGrade()
Input_Grade & Cal_Grade()
InFile_Close()
Show_Grade()
Identify
+Input_Password()
+Check_Password()
+Change_Password()
FileIO
+InFile_Open()
+InFile_Close()
+OutFile_Open()
+OutFile_Close()
+Set_FileName()
+Get_FileName()
Grade
+Input_Grade()
+Read_FGrade()
+Cal_Grade()
+Show_Grade()

v. 일정관리


휴강, 보강, 연강처리
사용자
사용자인증
일정관리
<<include>>
과목검색
<<extend>>
퀴즈, 과제 안내
<<extend>>
날짜검색
<<extend>>
출석관리
패스워드입력


: 사용자
: 사용자 인증
DataBase접근
일정관리
Input_Password()
Check_Password()
InFile_Open & OutFile_Open()
변경처리method()
InFile,Close & OutFIle_Close()


Identify
+Input_Password()
+Check_Password()
+Change_Password()
FileIO
+InFile_Open()
+InFile_Close()
+OutFile_Open()
+OutFile_Close()
+Set_FileName()
+Get_FileName()
Date
+Input_StartDate()
+Set_Subject()
+Input_Date()
+Show_Schedule()
+Show_Menu()
+Add_Schedule()
+Add_ASubject()
+Del_ASubject()
+Extend_ASubject()
+Check_Attend()

② Glossary(Contracts)

ⅰ. Identify Class : 사용자 인증과정과 관련

-Input_Password() :

사용자로부터 비밀번호를 입력받아 저장한다.

-Check_Password() :

사용자가 입력한 비밀번호가 저장되어 있는 비밀번호와 같은지 확인한다.

-Change_Password :

기존의 비밀번호를 새로운 비밀번호로 변경한다.

ⅱ. FileIO Class : 파일의 인 & 아웃과 관련

'Class Mate'는 텍스트파일을 이용하여 정보를 읽기 및 쓰기하는 시스템이다. 따라서 파일 입,출력을 다루는 class를 별개로 만들어 다른 클래스들이 상속받을 수 있도록 한다.

-InFile_Open() :

읽고자하는 파일을 오픈하고 에러 발생 확인을 한다.

-InFile_Close() :

파일 읽기가 끝난 후 파일스트림을 닫는다.

-OutFile_Open() :

출력하고자 하는 파일을 오픈하고 에러 발생 확인을 한다.

-OutFile_Close() :

파일에 출력이 끝난 후 파일스트림을 닫는다.

-Set_FileName() :

다루고자 하는 입력 및 출력 텍스트파일의 이름을 설정한다.

-Get_FileName() :

설정된 텍스트파일의 이름을 반환한다.

ⅲ. SubManager Class : 과목 삭제 및 변경과 관련

-Delete_Subject() :

사용자로부터 삭제하고자하는 과목명을 입력받은 후 저장되어 있는 과목의 정보를 삭제한다.

-Change_SubInfo() :

이미 저장되어 있는 과목의 정보를 변경한다.

iv. Subject Class : 과목정보와 관련

-Add_SubInfo() :

사용자로부터 과목정보를 입력받아 저장한다.
(과목정보 : 과목명, 교수, 수강요일 및 시간,
강의장소, 학점(몇 학점 과목인지))

-Show_List() :

저장된 모든 과목의 정보를 출력한다.

v. Grade Class : 성적관리와 관련

-Input_Grade() :

사용자로부터 과목명과 성적을 입력받아서 저장한다.

-Read_FGrade() :

사용자가 입력한 과목이 몇 학점짜리 과목인지 DB에서
읽어온다.

-Cal_Grade() :

사용자로부터 입력받은 성적과 Read_FGrade()로
읽어들인 전체 학점을 이용하여 평점을 계산한다.

-Show_Grade() :

계산된 평점을 출력한다.

vi. Date Class : 일별 수강과목 관리 및 일정관리와 관련

-Input_StartDate() :

사용자로부터 개강일의 월, 일, 요일을 입력받아서
저장한다.

-Set_Subject() :

텍스트파일에 저장된 과목정보 Database를 바탕으로
학기 중 일별로 수강해야 할 과목을 세팅한다. 이 후,
파일입력 메소드를 이용하여 세팅이 완료된
수업정보로 새로운 Database를 작성한다.

-Input_Date() :

사용자로부터 조회하고 싶거나 변경하고 싶은 날짜를
입력받는다.

-Show_Schedule() :

사용자가 입력한 날의 일정 및 출석상태를 출력한다.

-Show_Menu() :

일정에 관련된 메뉴를 출력한다.
1. 과제 및 퀴즈 등의 이벤트 추가
2. 보강, 휴강 등 해당 일의 과목 추가 및 삭제
3. 출석관리

-Add_Schedule() :
사용자가 조회한 날짜에 과제, 시험 등의 정보를 추가한다.

-Add_ASubject() :
보강 일정이 생겼을 경우 해당 날짜에만 보강된 수업을 추가한다.

-Del_ASubject() :
휴강 일정이 생겼을 경우 해당 날짜에만 휴강된 수업을 삭제한다.

-Extend_ASubject() :
연강 일정이 생겼을 경우 해당 날짜의 연강된 과목의 강의시간을 늘린다.

-Check_Attend() :
해당 날짜에 수강하는 과목들의 출석상태를 변경한다. (boolean변수를 이용)