CLASS MATE

User-friendly Timetable program

TEAM 9
200911385 박기남
200911425 조서경
200911426 조성완
200911427 조아라*

INDEX

Introduction

- 목적

-Practicality / Convenience
: 조작이 간편하고 시간표 관리 이외의
다양한 기능을 수행

-Visibility
: 한 눈에 들어올 수 있도록 메뉴 , 일정, 시간표 출력

-Safety
: 암호를 이용한 사용자 인증

Introduction

▶ 목적(Cont.)

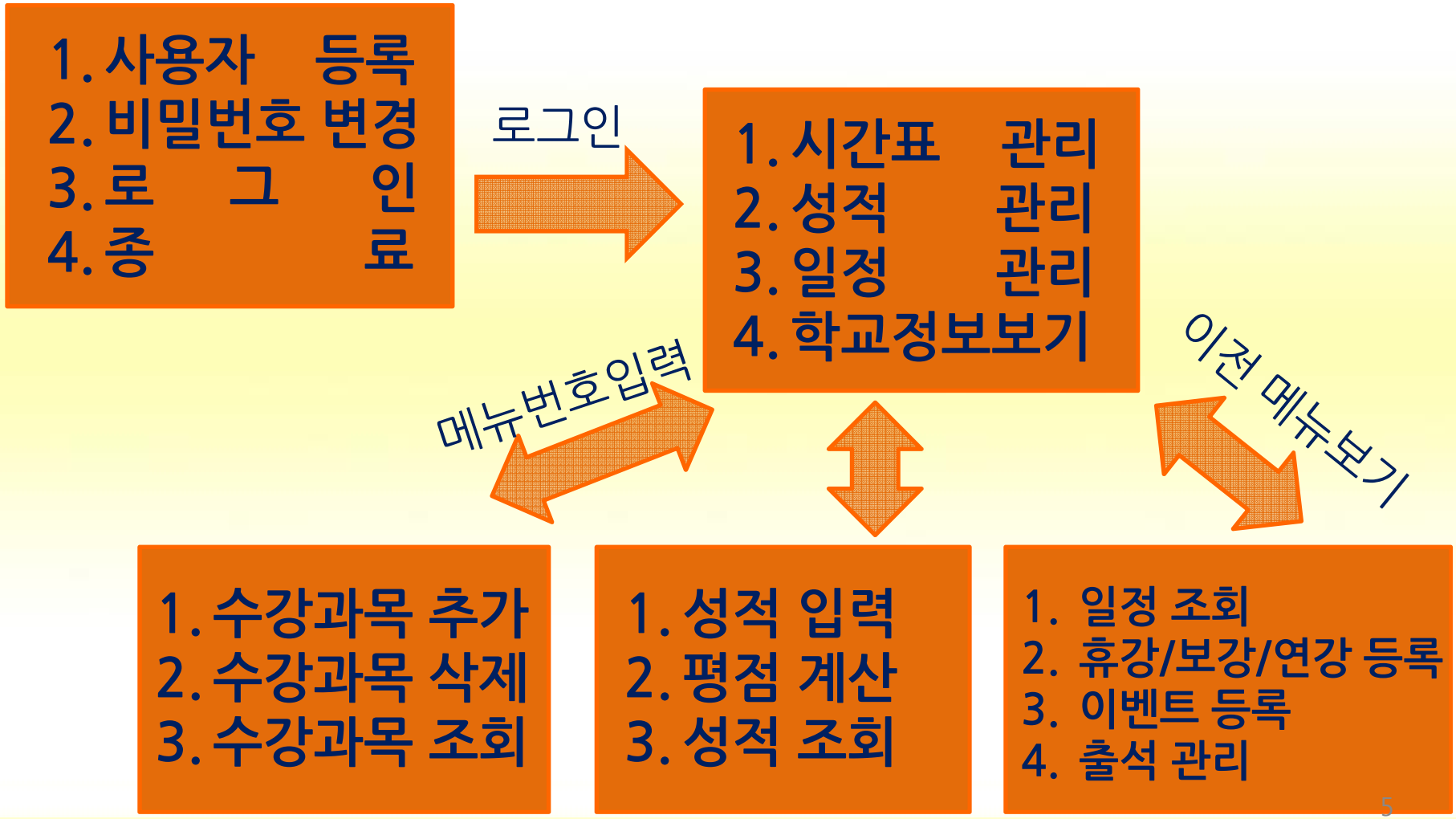
- User friendly

: 원하는 기능의 메뉴 번호를 선택하여 프로그램을 실행
→ 쉬운 조작으로 누구나 쉽게 사용할 수 있다.

: 일반적인 기능 외 출석, 성적 관리 및
도서관 열람실 현황조회/ 교내시설 전화번호 검색기능 추가
→ 사용자의 일상에서 보다 자주, 유용하게 쓰일 수 있도록 하였다.

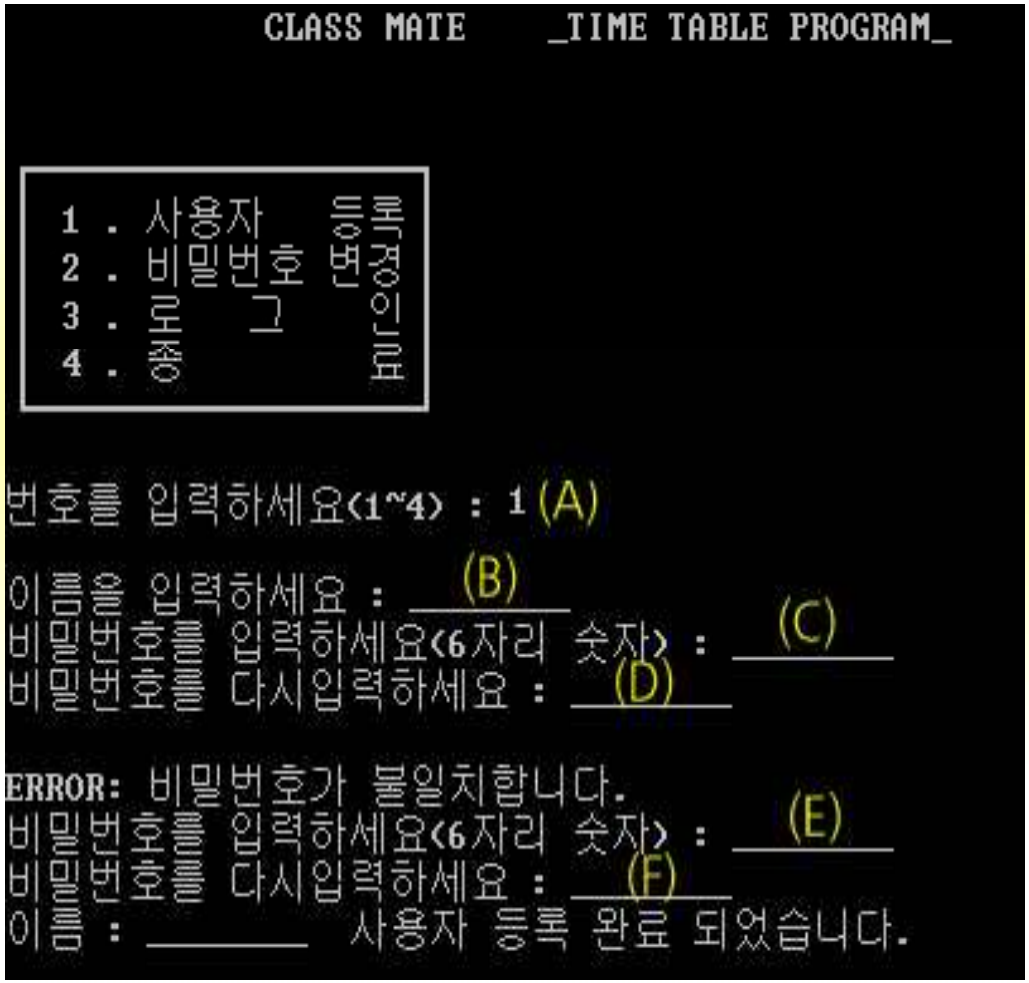
Introduction

▶ 주요 메뉴 및 프로세스



Real Use-case

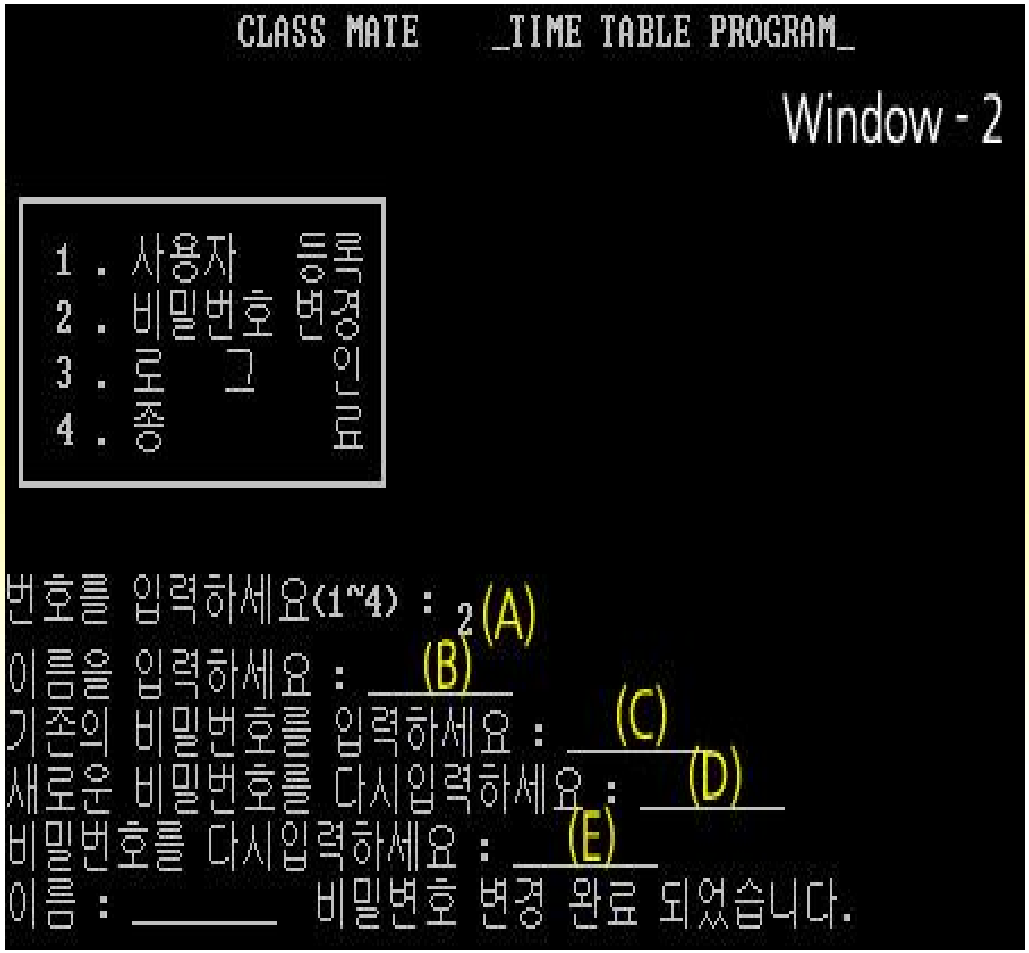
▶ 사용자 등록 : 처음 사용하는 사용자의 **정보 등록**



Actor	System
'사용자등록' 메뉴선택(A) →	이름, 비밀번호 입력 요청
B, C, D에 정보 입력 →	사용자 DB에 정보 저장

Real Use-case

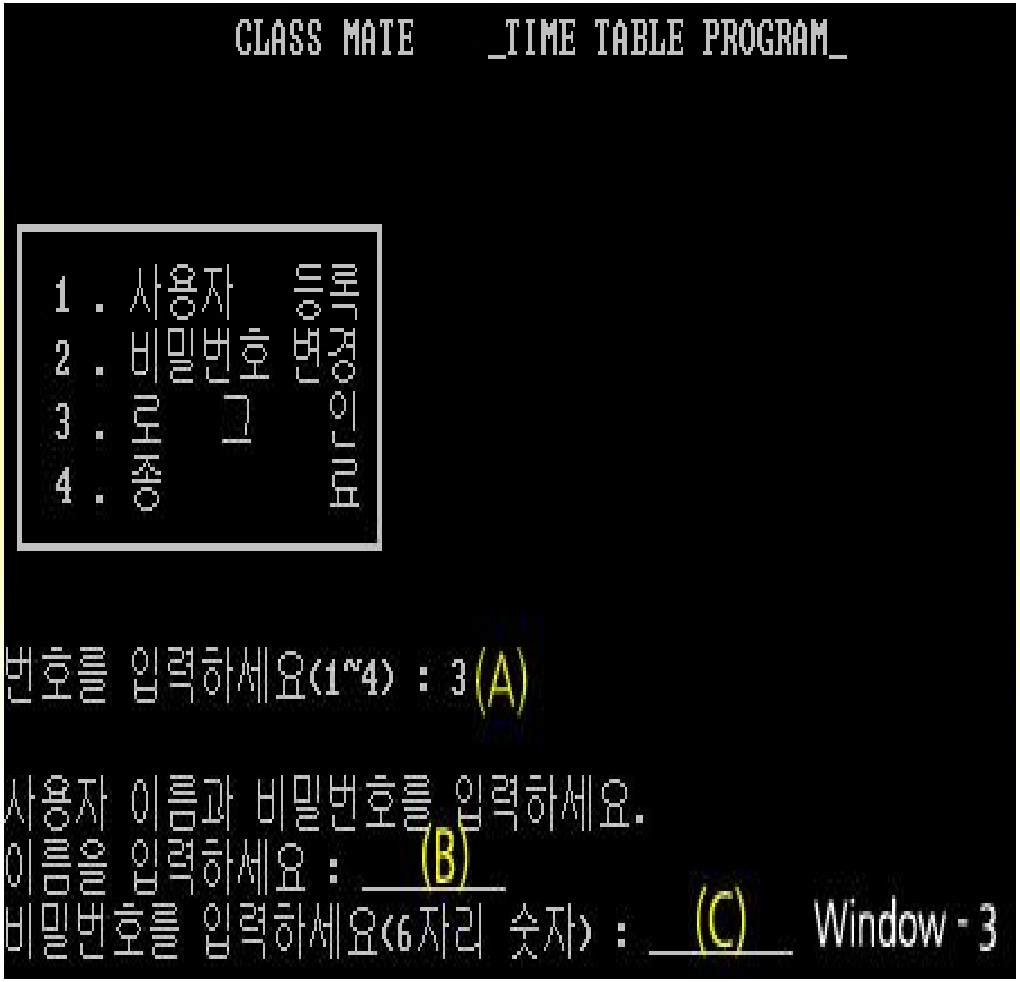
▶ 비밀번호 변경



Actor	System
'비밀번호 변경' 메뉴선택(A)	→ 이름, 비밀번호 입력 요청
B, C에 정보 입력	→ 사용자 인증 확인 새로운 비밀번호 요청
D,E에 새로운 비밀번호 입력	→ 사용자 DB에서 비밀번호 변경 후 저장

Real Use-case

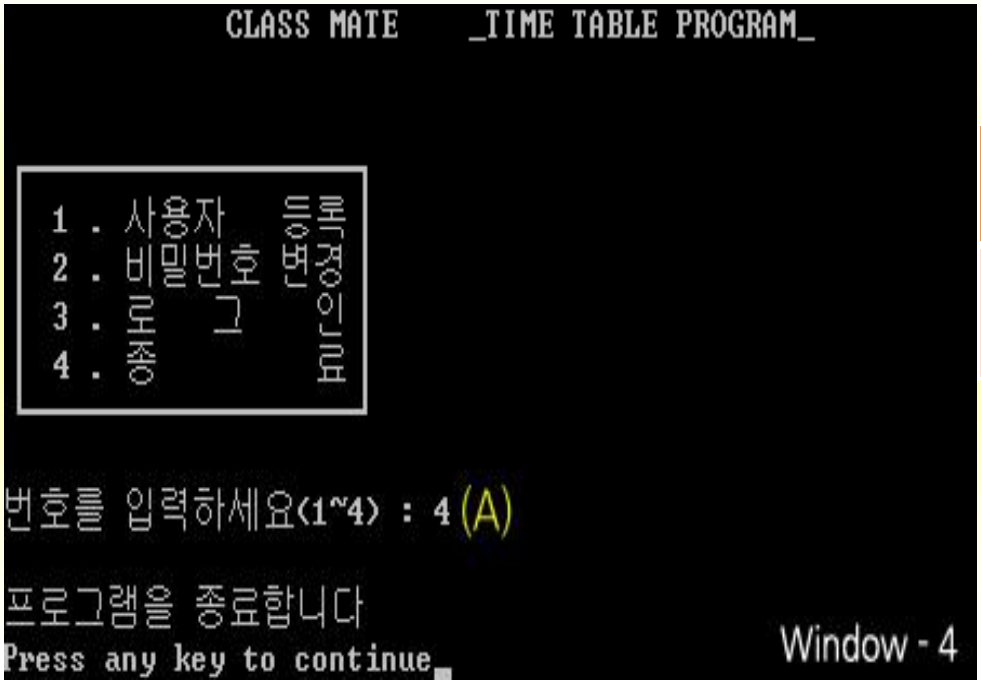
▶ **사용자 인증:** 이름, 비밀번호 입력 → **자기 자신 인증**



Actor	System
'로그인' 메뉴선택(A) →	← 이름, 비밀번호 입력 요청
B, C에 정보 입력 →	← 입력정보 인증 후 메인 메뉴출력

Real Use-case

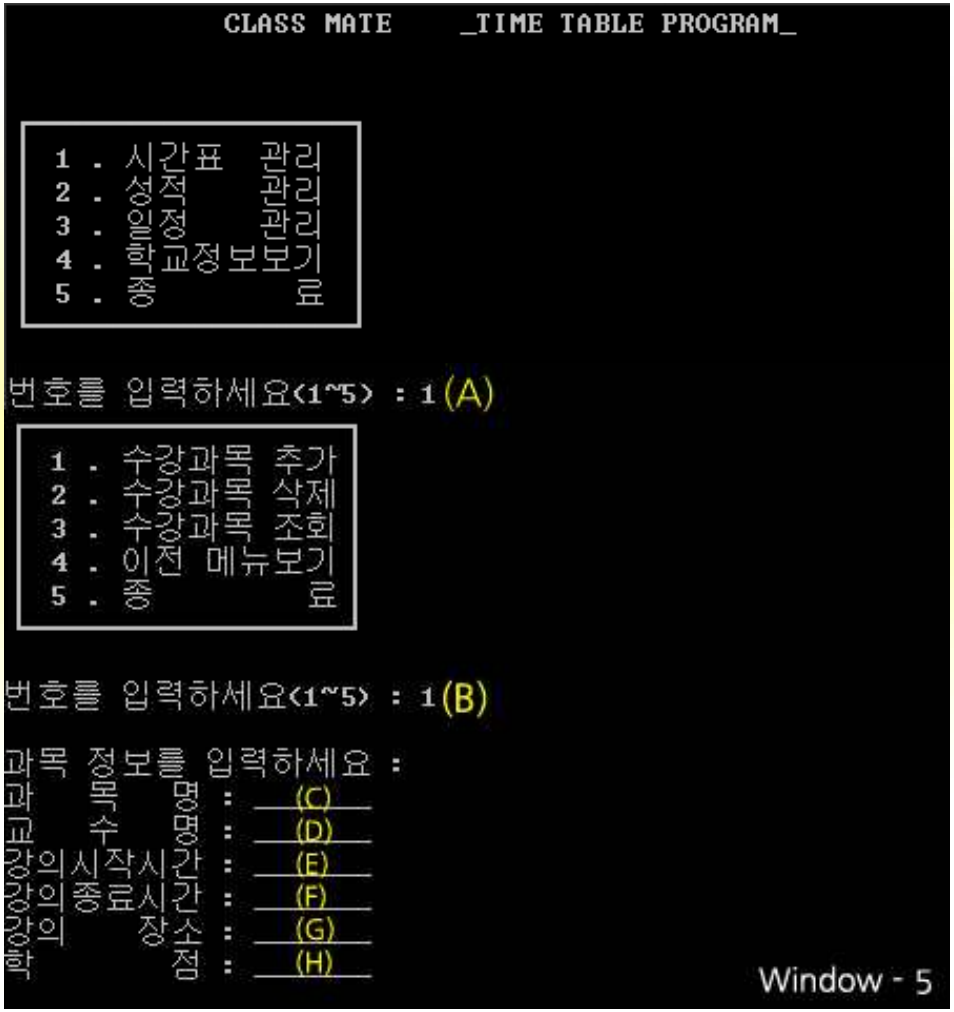
▶ 프로그램 종료



Actor	System
'종료' 메뉴선택(A) →	종료 메시지 출력 프로그램 종료

Real Use-case

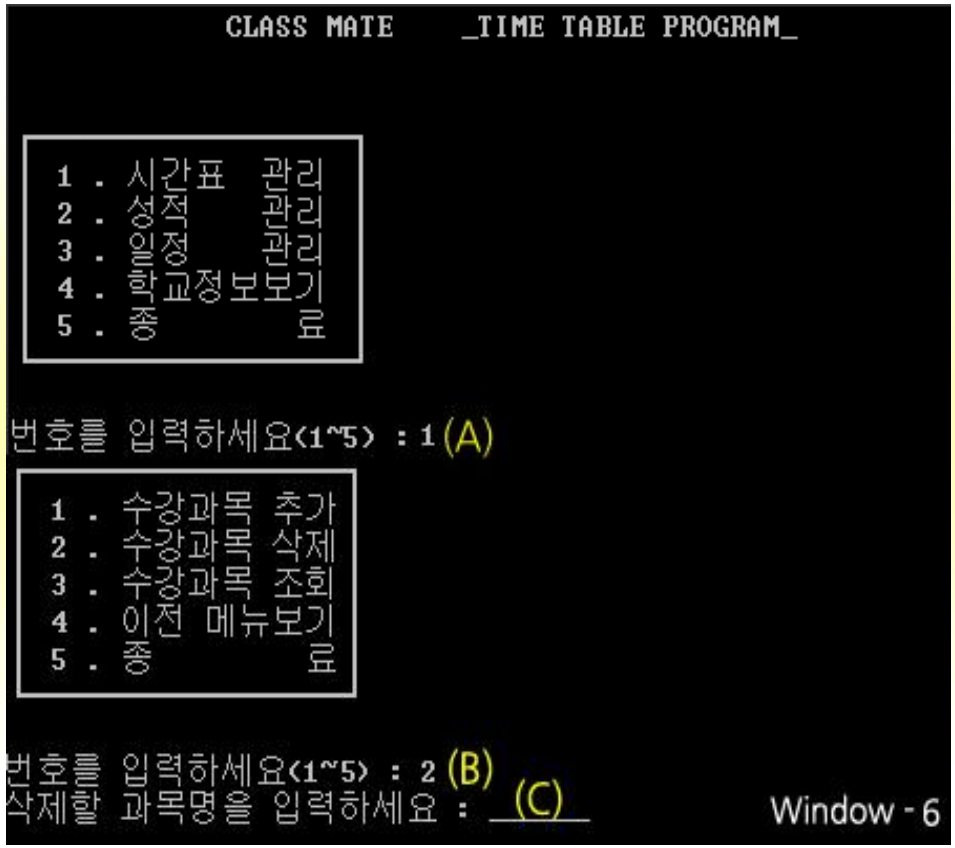
▶ 수강과목 추가: 사용자의 수강 과목을 시간표에 추가



Actor	System
'시간표 관리' 선택(A) -> '과목 추가' (B) →	과목정보 입력 요청
C~H에 과목정보 입력 →	입력 받은 정보를 과목정보 DB에 저장

Real Use-case

▶ 수강과목 삭제: 사용자가 시간표에 추가했던 과목을 삭제



Actor	System
'시간표 관리'선택(A) -> '과목 삭제'선택(B)	삭제할 과목명 입력 요청
C에 과목명 입력	입력 받은 과목의 정보를 DB에서 삭제

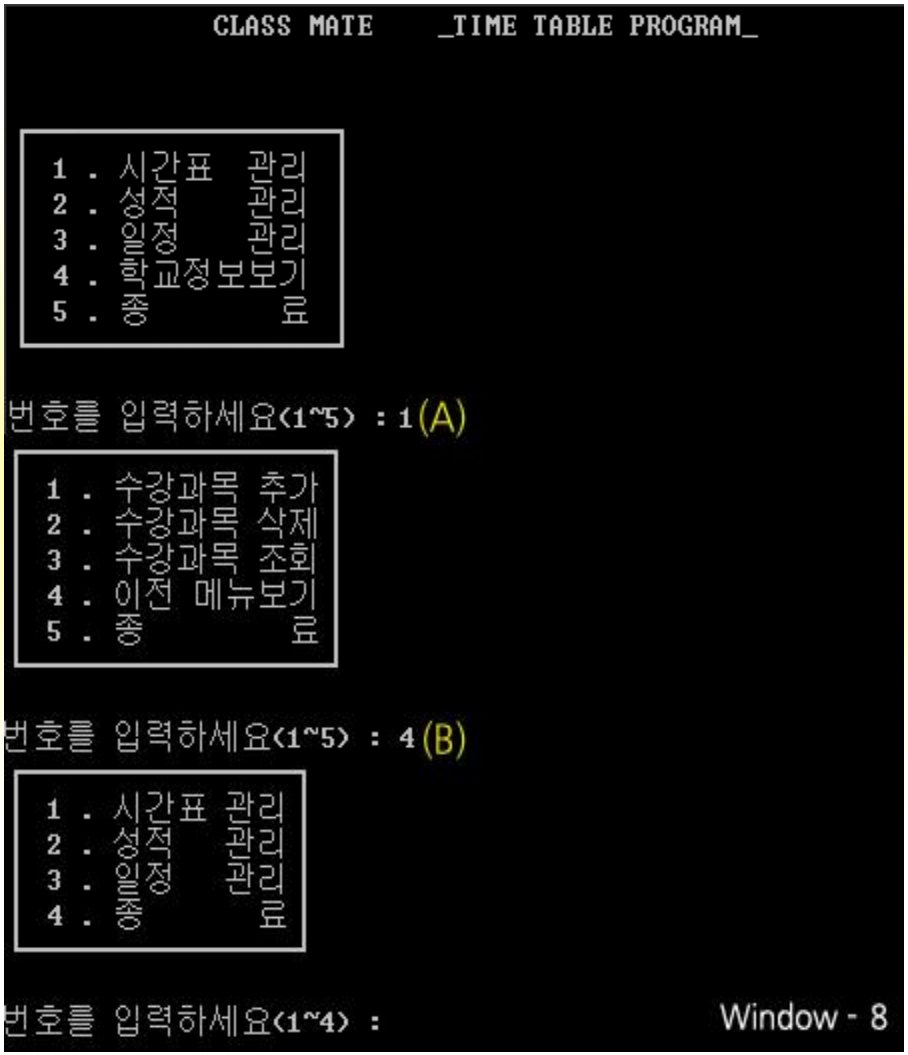
Real Use-case

▶ 수강과목 조회: 사용자가 시간표에 추가한 과목을 조회

Actor	System
'시간표관리'선택(A) ->'과목 조회'선택(B)	현재 수강중인 과목출력

Real Use-case

▶ 이전 메뉴보기: 상위 메뉴를 조회

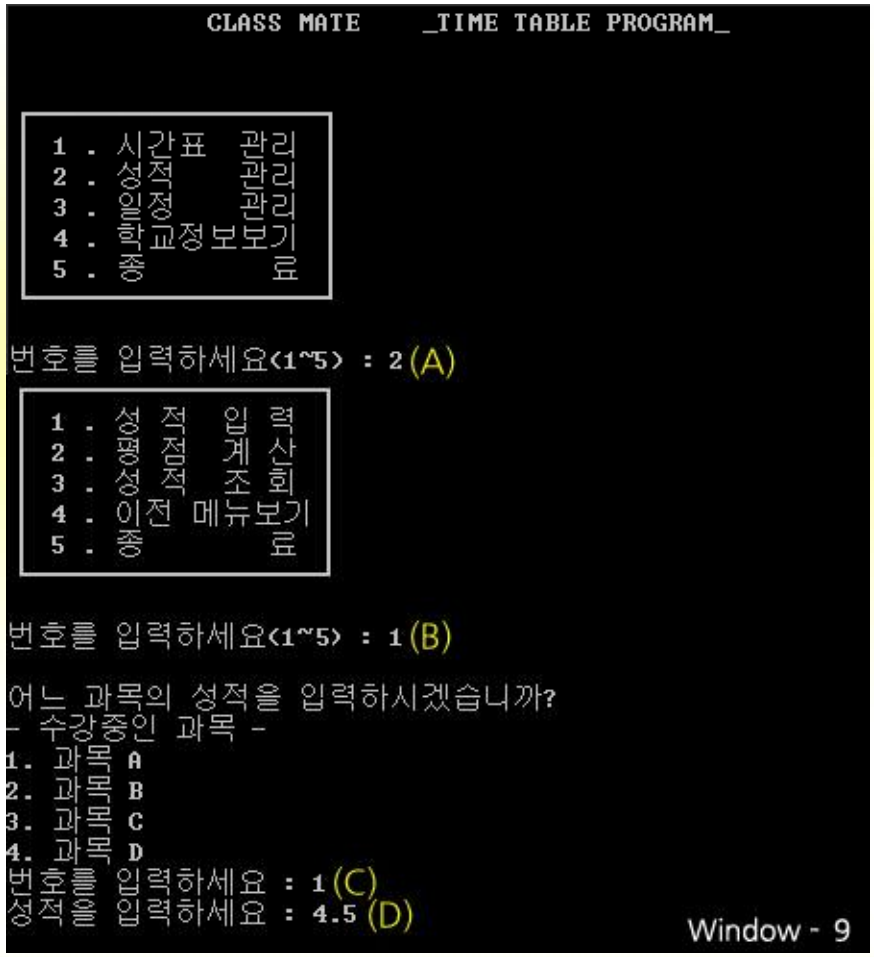


Actor	System
모든 서브메뉴 에서 ‘이전 메뉴보기’ 메뉴 선택(A)(B) →	상위 메뉴인 시간표/성적/일정 관리 메뉴를 재 출력

Real Use-case

▶ 성적 관리: 성적을 입력, 조회하고 평점 계산 후 출력

- 성적 입력



Actor	System
'성적 관리'선택(A) -> '성적 입력'선택(B)	수강중인 과목 리스트 출력 성적 입력할 과목번호 요청
C,D에 과목번호, 성적 입력	입력 받은 성적 정보를 DB에 저장

Real Use-case

▶ 성적 관리 (cont.)

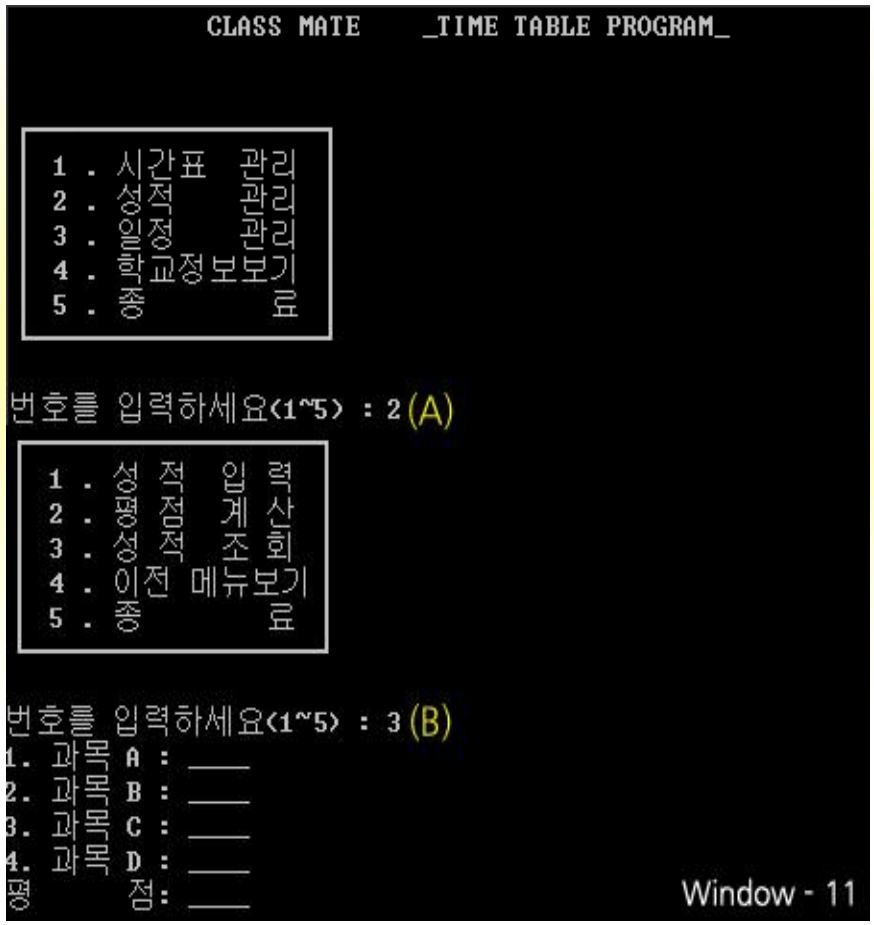
- 평점 계산

Actor	System
'성적 관리'선택(A) -> '평점 계산'선택(B)	성적정보를 DB에서 읽어와서 평점을 계산, 출력

Real Use-case

▶ 성적 관리 (cont.)

- 성적 조회

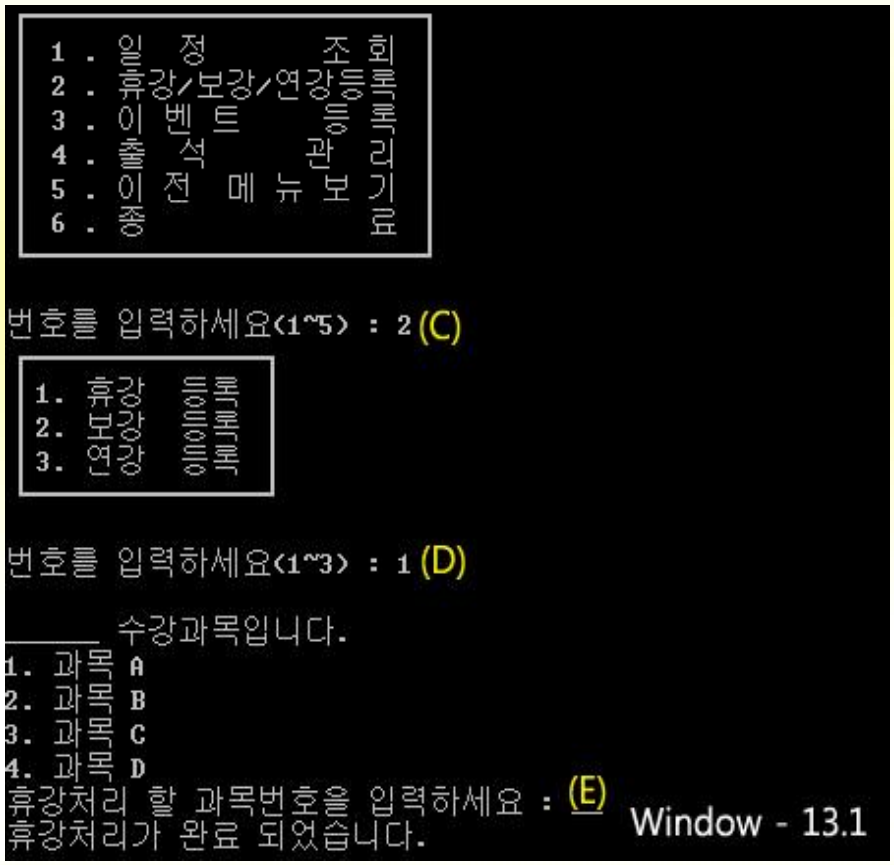


Actor	System
'성적 관리'선택(A) -> '성적 조회'선택(B)	성적정보를 DB에서 읽어와서 평점과 함께 출력

Real Use-case

▶ 일정 관리: 사용자가 수강하고 있는 과목의 일정을 조회, 관리
(휴강, 보강, 연강 등의 일정 등록/ 이벤트 정보 입력/ 과목별 출석상태 관리)

- 휴강, 보강, 연강처리(예 : 휴강 등록)



Actor	System
'일정 관리'선택(A) -> 관리할 날짜 선택(B) -> '휴강/보강/연강등록'선택(C) ->'휴강등록' 선택(D)	수강중인 과목 리스트 출력 휴강처리 할 과목번호 요청
E에 과목번호 입력	DB에서 해당날짜의 해당 과목정보를 수정하여 저장

Real Use-case

▶ 일정 관리 (cont.)

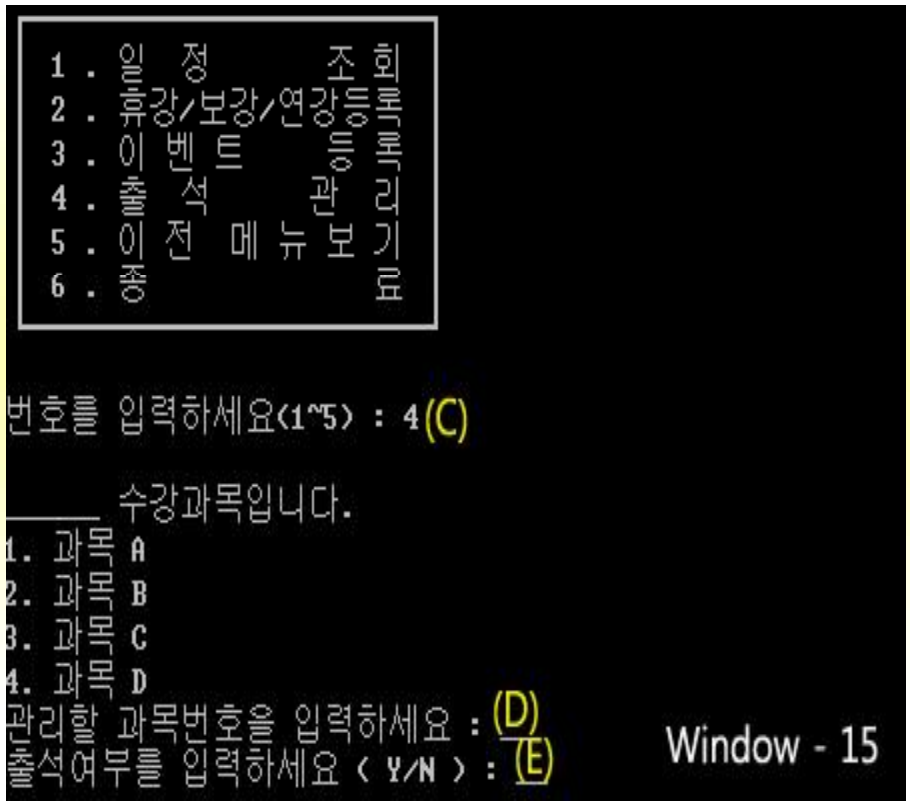
- 이벤트 등록

Actor	System
'일정 관리'선택(A) -> 관리할 날짜 선택(B) -> '이벤트 등록'선택(C)	수강중인 과목 리스트 출력 관리할 과목의 번호를 요청
D에 관리하고자 하는 과목 번호 입력	이벤트 정보 입력 요청
E~H에 이벤트 정보 입력	DB에서 해당날짜의 해당 이벤트 정보를 저장

Real Use-case

▶ 일정 관리 (cont.)

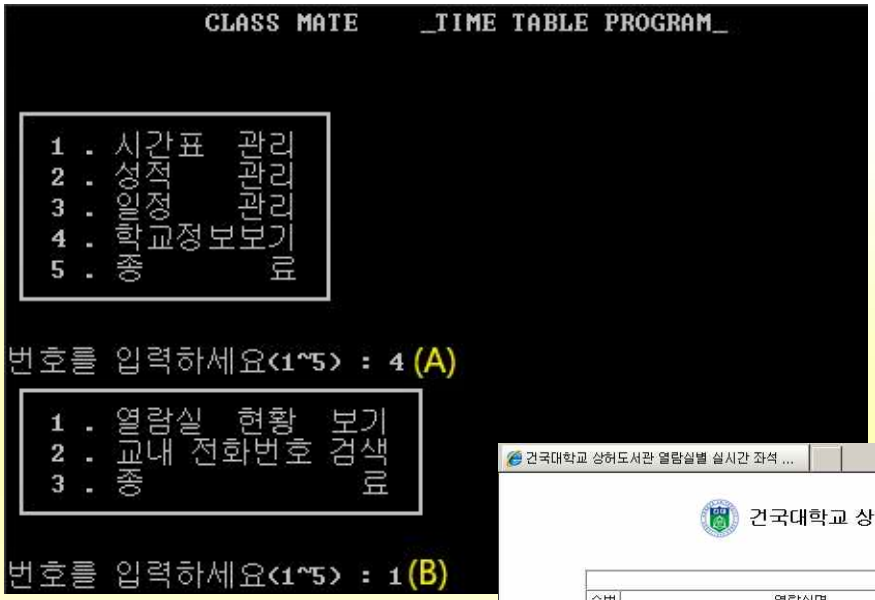
- 출석 관리



Actor	System
'일정 관리'선택(A) -> 관리할 날짜 선택(B) -> '출석관리' 선택(C) →	수강중인 과목 리스트 출력 출석관리 할 과목번호 요청
D,E에 과목번호, 출석여부 입력 →	해당날짜, 해당 과목의 DB에 출석상태를 저장함

Real Use-case

▶ 열람실 현황 보기: 사용자가 현재 도서관 열람실 정보를 조회

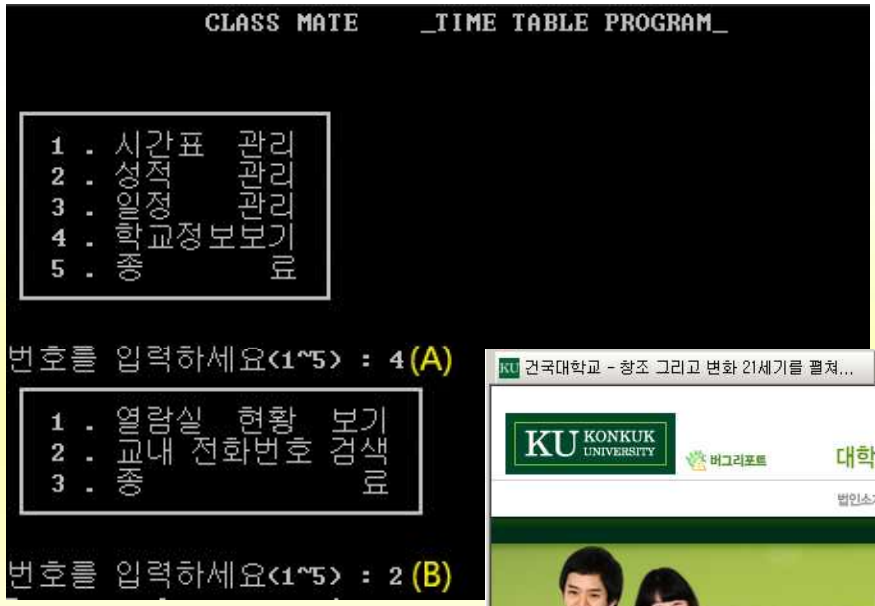


Actor	System
'학교정보보기'선택(A) -> '열람실 현황보기' 선택(B)	→ 도서관 열람실 현황 웹페이지 링크 오픈

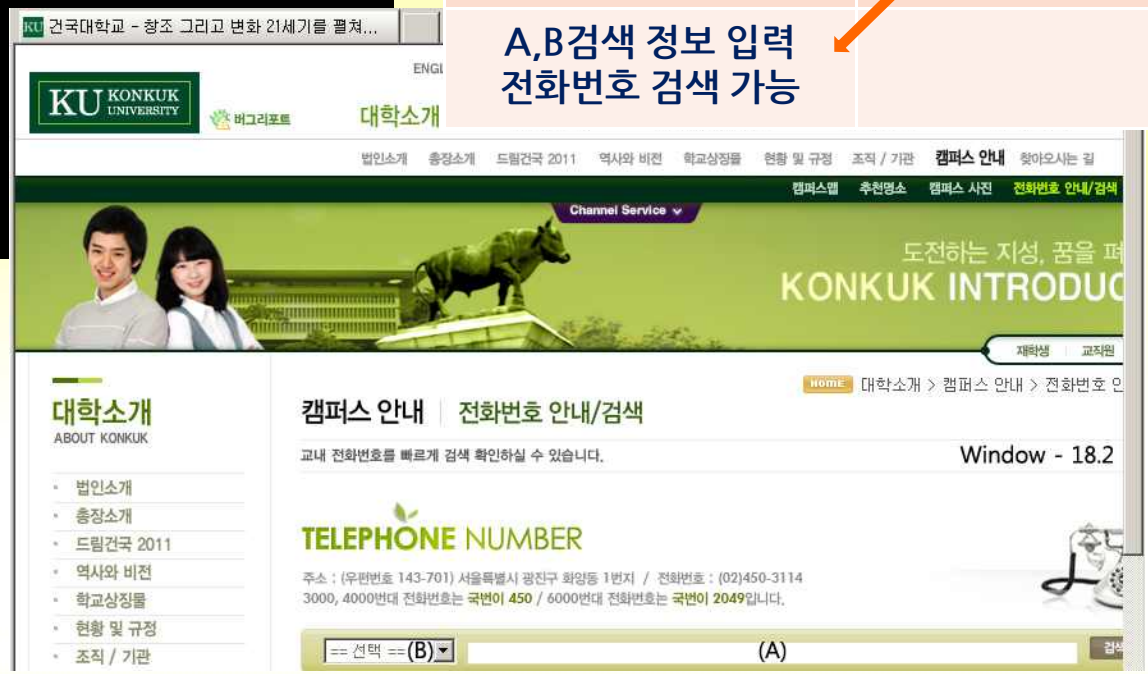
순번	열람실명	전체 좌석수	사용 좌석수	잔여 좌석수	이용율(%)
1	제1A 열람실	300	165	135	55 %
2	제1B 열람실	294	124	170	42.18 %
3	제2 열람실	184	71	113	38.59 %
4	제3A 열람실	332	120	212	36.14 %
5	제3B 열람실	284	103	181	36.27 %
6	제4 열람실	172	65	107	37.79 %
7	제5 열람실	168	36	132	21.43 %
8	제6 열람실	252	129	123	51.19 %
9	국가고시연구실	100	0	100	0 %
10	6층노트북열람실	72	45	27	62.5 %
11	1층노트북열람실	46	36	10	78.26 %
12	-	1	0	1	0 %

Real Use-case

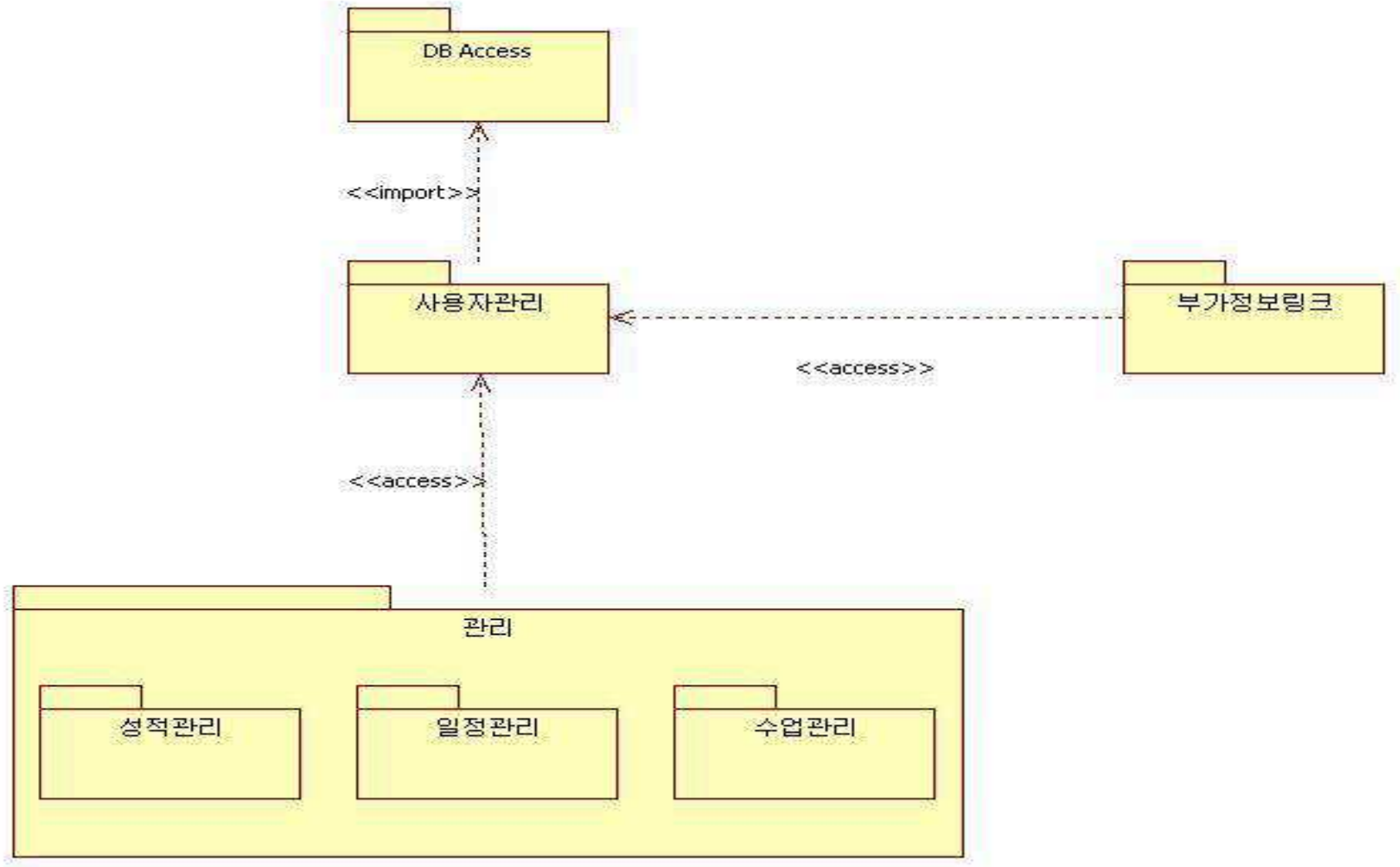
▶교내 전화번호 검색: 사용자가 교내 시설 전화번호 검색



Actor	System
'학교정보보기'선택(A) -> '교내 전화번호 검색' 선택(B)	교내 시설 전화번호 검색이 가능한 웹페이지 링크 오픈
A,B검색 정보 입력 전화번호 검색 가능	

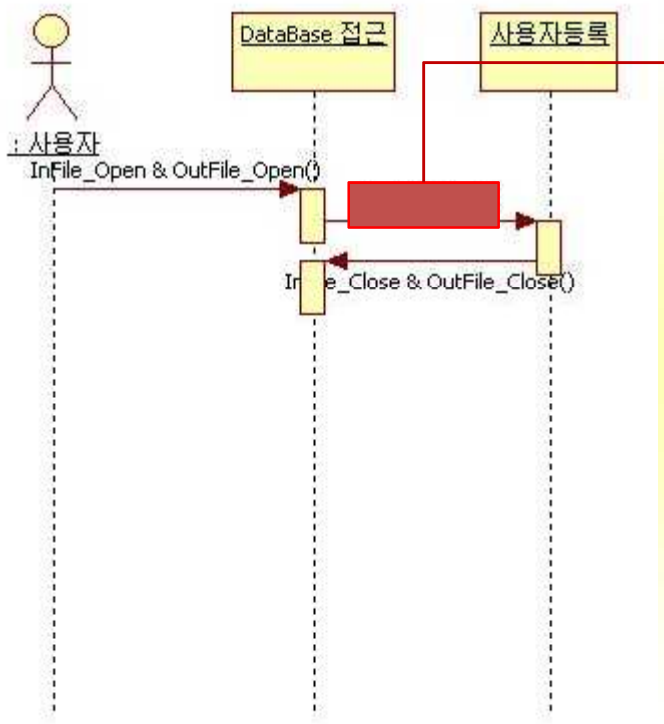


Package diagram



Interaction Diagram(Sequence diagram)

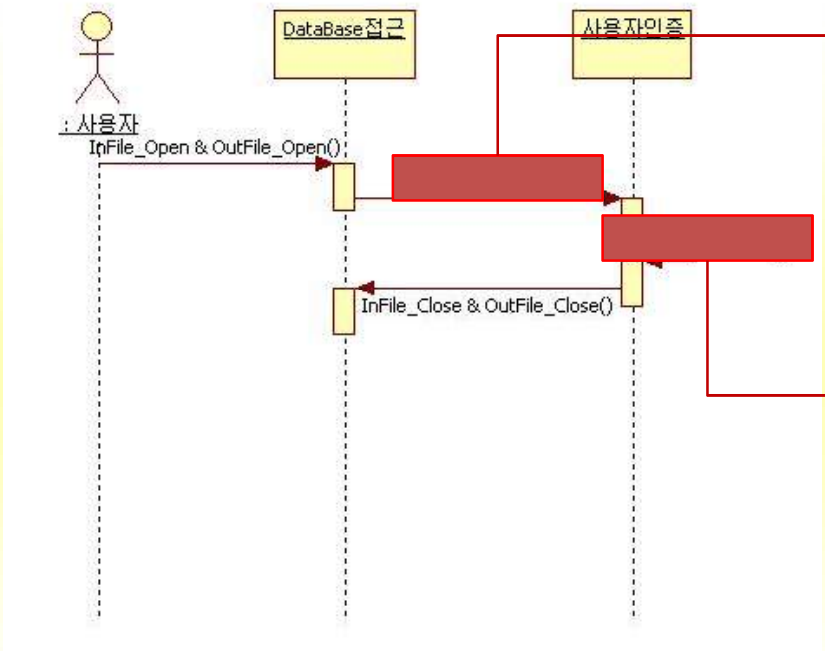
▶ 사용자등록



Identify::Regist_User()
⇨ DB에 새로운 사용자의 정보를 추가

Interaction Diagram(Sequence diagram)

▶ 사용자인증

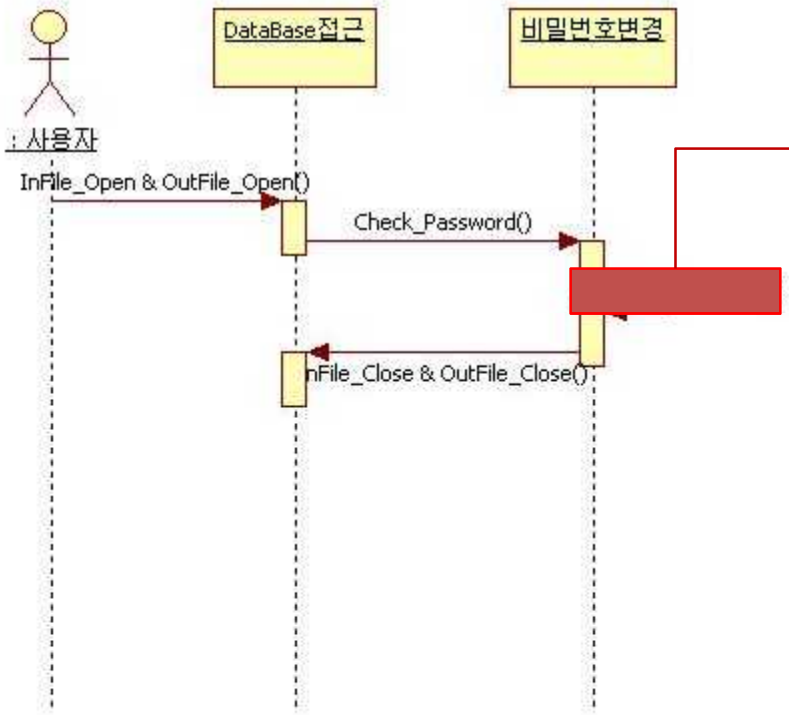


Identify::Input_ID()
⇨ 사용자이름, 비밀번호를 입력 받아 Buffer에 저장

Identify::Check_Password()
⇨ 입력된 정보가 맞는지 검사

Interaction Diagram(Sequence diagram)

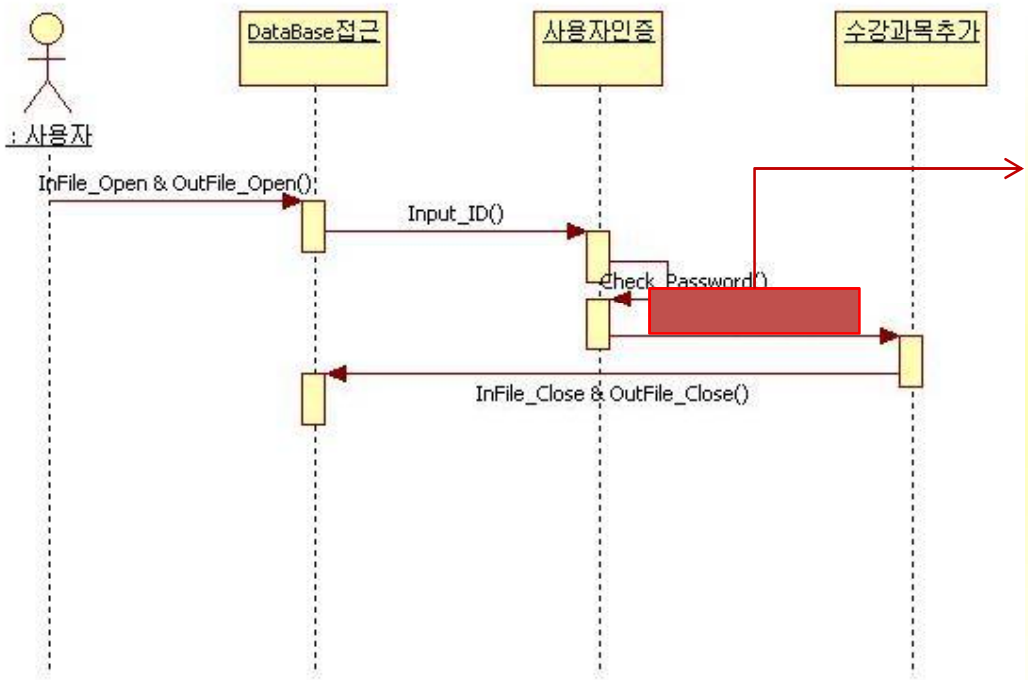
▶ 비밀번호변경



Identify::Change_Password()
⇨ DB에 변경된 사용자 비밀번호를 적용

Interaction Diagram(Sequence diagram)

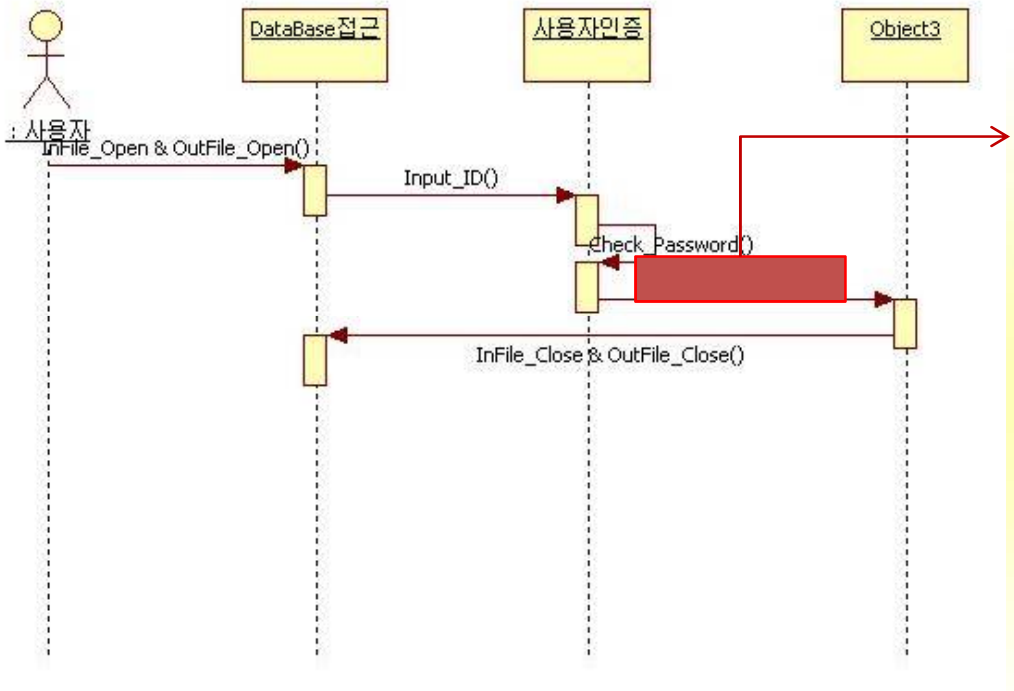
▶ 수강과목추가



Subject::Add_SubInfo()
⇒ 과목정보를 입력받아 DB에 적용

Interaction Diagram(Sequence diagram)

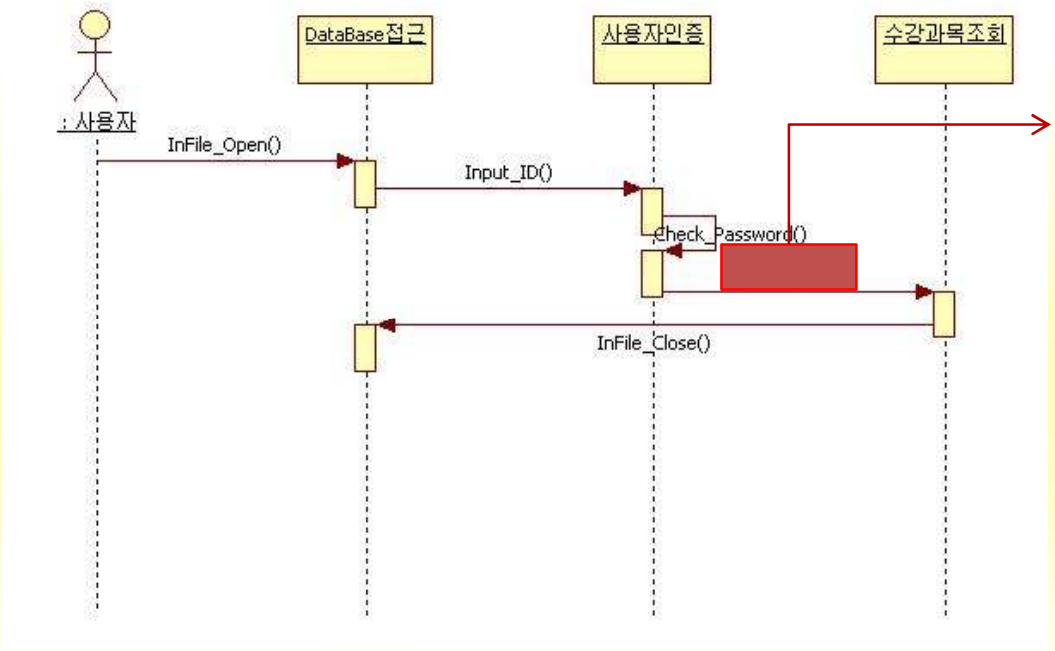
▶ 수강과목삭제



SubManager::Delete_Subject()
⇨ 과목명 입력받아 DB에 적용

Interaction Diagram(Sequence diagram)

▶ 수강과목조회

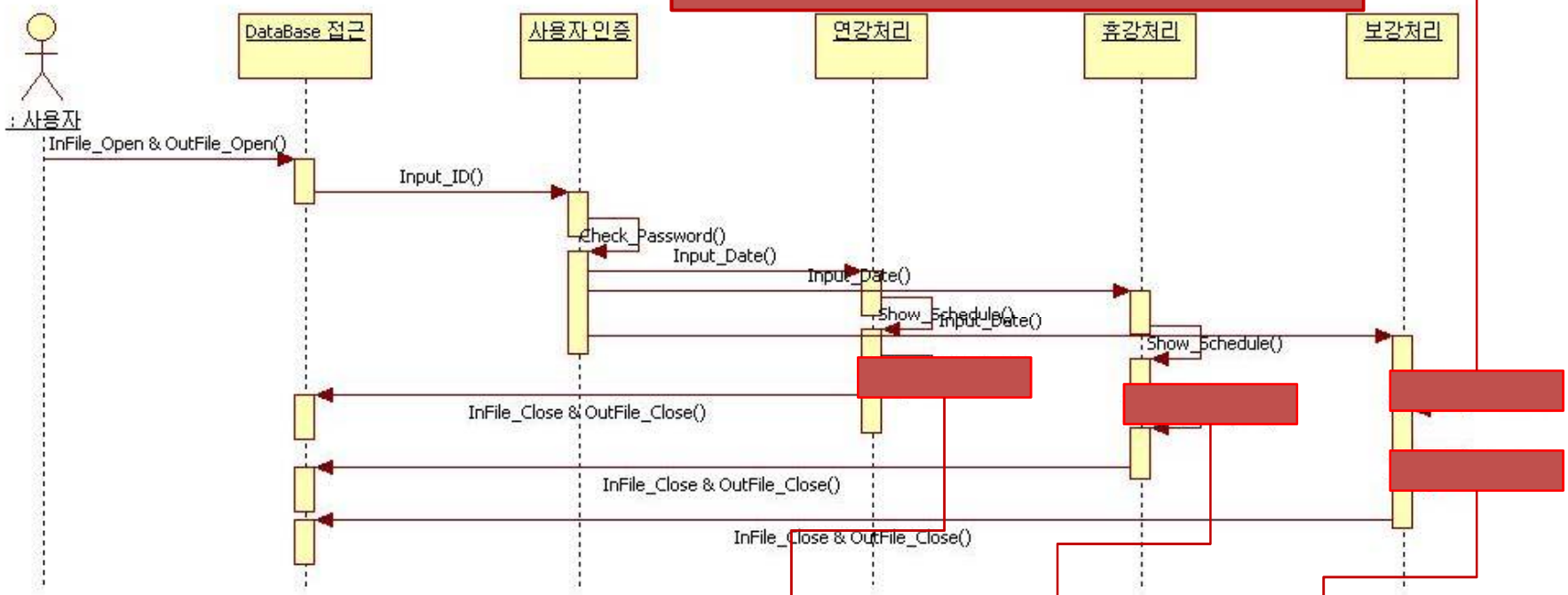


SubManager::Show_List()
⇨ 현재 수강중인 과목List를 실행화면에 출력

Interaction Diagram(Sequence diagram)

▶ 일정관리

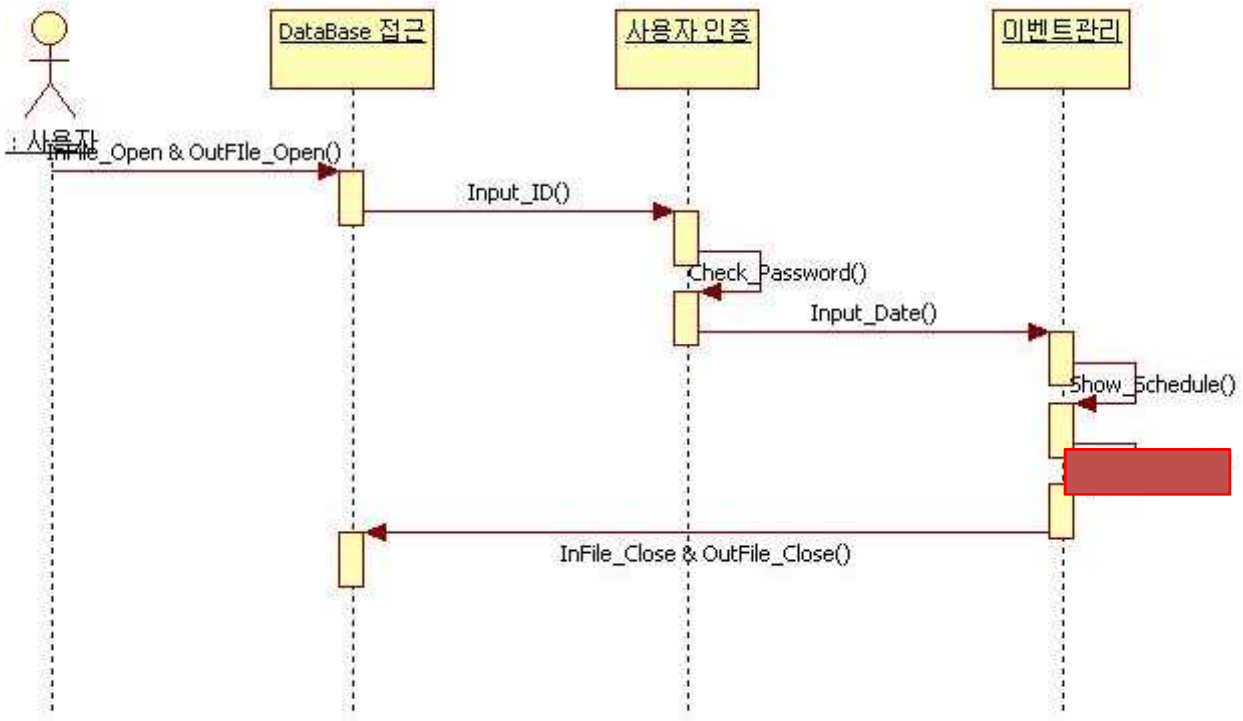
Date::Show_Schedule()
⟹ 날짜를 입력받아 해당일의 일정을 실행화면에 출력



Date::Extend(Del, Add)_ASubject(),
⟹ 날짜와 과목을 입력받아 DB에 변경사항 적용

Interaction Diagram(Sequence diagram)

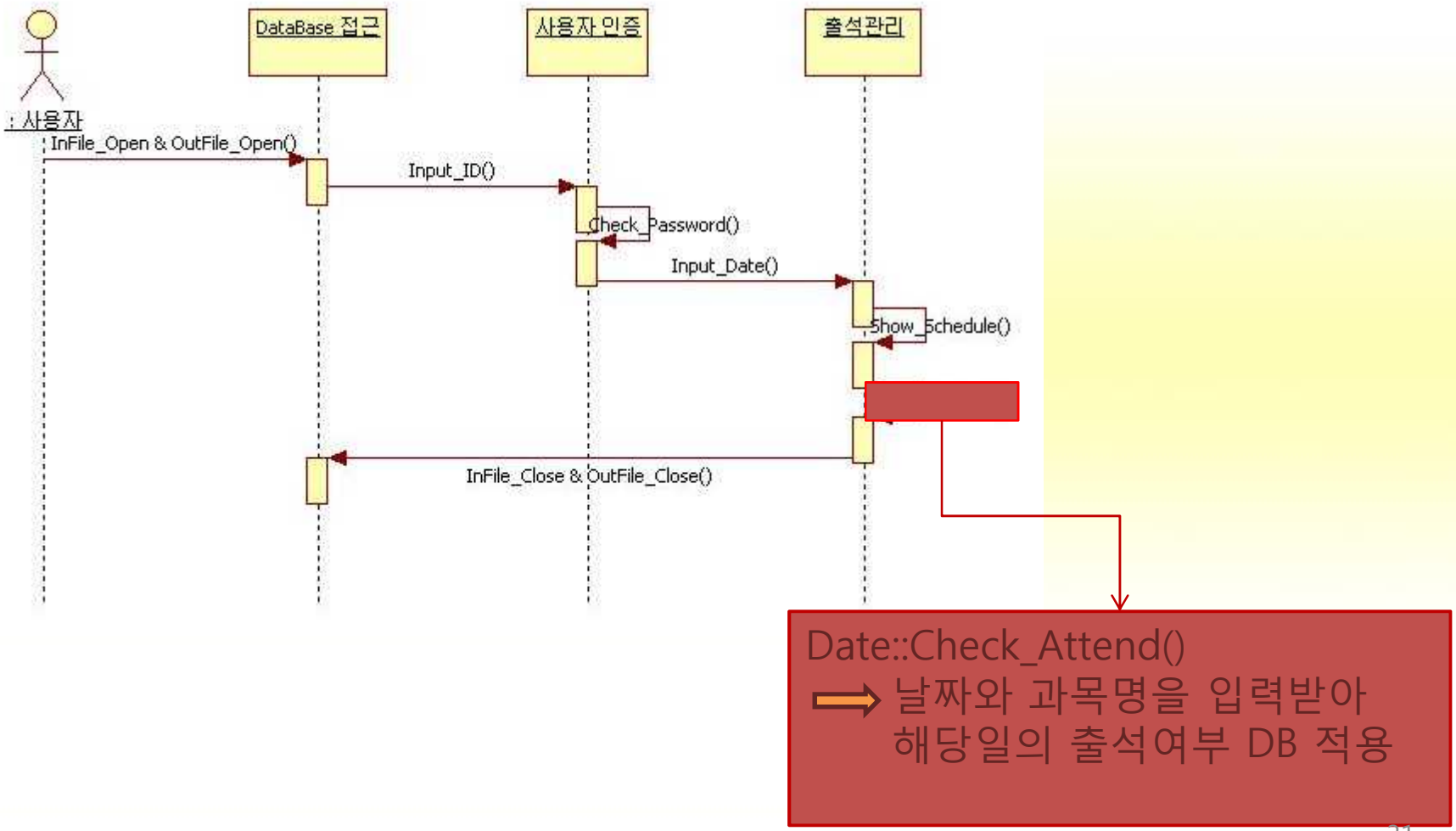
▶ 일정관리 (cont.)



Date::Add_Schedule
⇨ 날짜를 입력받아 이벤트를 DB에 적용

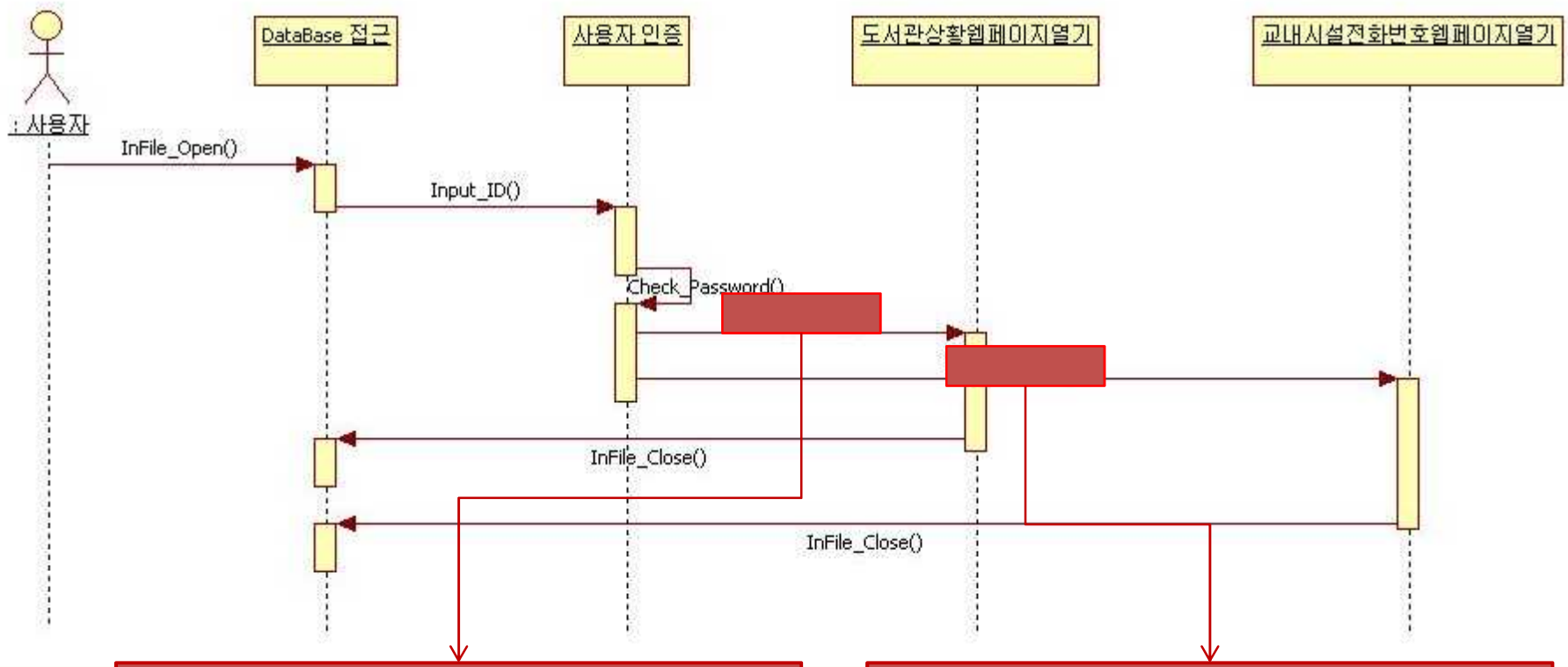
Interaction Diagram(Sequence diagram)

▶ 일정관리 (cont.)



Interaction Diagram(Sequence diagram)

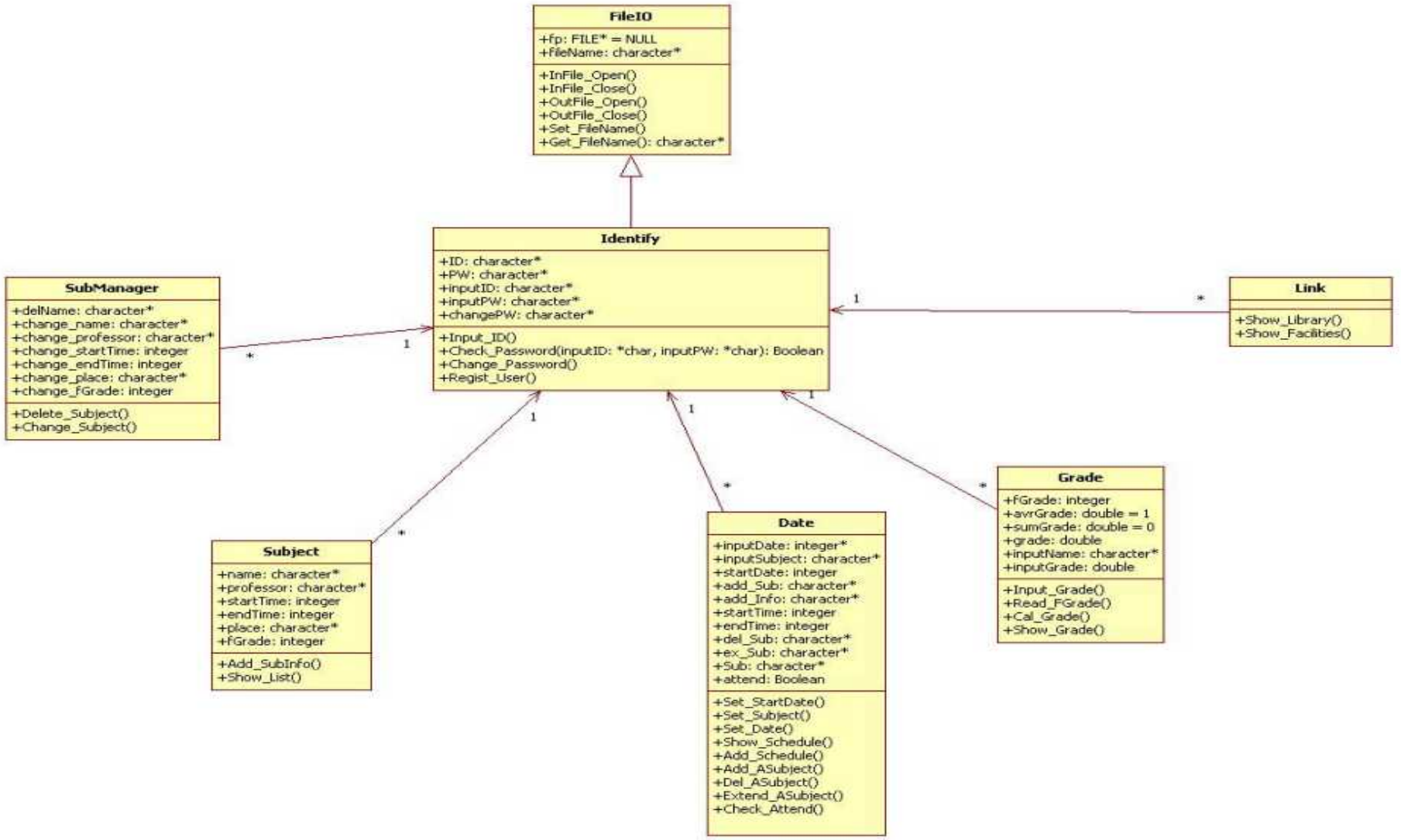
▶ 링크



Link::Show_Library()
⇨ 도서관 좌석현황의 웹페이지를 링크오픈

Link::Show_Facilities()
⇨ 교내시설 전화번호부 웹페이지를 링크오픈

A Design Class Diagram



DataBase Schema

▶ Class Identify

```
char *ID;                          // 파일에서 읽어온 ID Buffer
char *PW;                          // 파일에서 읽어온 PW Buffer

void Input_ID()
        char *inputID;
        char *inputPW;             // 사용자가 입력한 비밀번호
bool Check_Password(char *inputID, char *inputPW)
void Change_Password()
        char *inputID;
        char *inputPW;
        char *changePW;            // 변경할 비밀번호
void Regist_User()
        char *inputID;
        char *inputPW;
```

Key name	Data type	NULL	Default value	Key information
ID	char *	X	DB에서 Load	파일에서 읽어온 ID
PW	char *	X	DB에서 Load	파일에서 읽어온 비밀번호
inputID	char *	X		사용자가 입력한 ID
inputPW	char *	X		사용자가 입력한 비밀번호
changePW	char *	X		바뀔 비밀번호

DataBase Schema

▶ Class FileIO

```
FILE *fp;                    // 파일 포인터
char *FileName;              // 파일이름 (저장 / 반환)

void InFile_Open()
void InFile_Close()
void OutFile_Open()
void OutFile_Close()
void Set_FileName()
char *Get_FileName()
```

Key name	Data type	NULL	Default value	Key information
FileName	char *	X	DB에서 Load	Database 텍스트파일의 이름
fp	FILE *	O	NULL	파일 포인터

DataBase Schema

Class SubManager

```
void Delete_Subject()
        char *delName;              // 지우고자 하는 과목의 과목명
void Change_Subject()
        char *change_name;          // 정보변경하고자 하는 과목의 과목명
                                    // 바꾸고자하는 세부정보
        char *change_professor;     // 담당교수명
        int change_startTime;       // 강의가 시작하는 시간
        int change_endTime;         // 강의가 끝나는 시간
        char *change_place;         // 강의실
        int change_fGrade;          // 학점단위
```

Key name	Data type	NULL	Default value	Key information
delName	char *	X		사용자가 지우고자하는 과목명
change_name	char *	X		과목정보를 변경할 과목의 이름
change_professor	char *	X		변경될 담당교수이름
change_startTime	int	X		변경될 강의 시작시간
change_endTime	int	X		변경될 강의 종료시간
change_place	char *	X		변경될 강의실
change_fGrade	int	X		변경될 학점단위

DataBase Schema

Class Subject

```
void Add_SubInfo()
        char *name;              // 과목명
        char *professor; // 담당교수명
        int startTime;           // 강의가 시작하는 시간
        int endTime;             // 강의가 끝나는 시간
        char *place;             // 강의실
        int fGrade;              // 학점단위
void Show_List()
```

Key name	Data type	NULL	Default value	Key information
name	char *	X		과목의 이름
professor	char *	X		담당교수이름
startTime	int	X		강의 시작시간
endTime	int	X		강의 종료시간
place	char *	X		강의실
fGrade	int	X		학점단위

DataBase Schema

Class Grade

```
int fGrade;              // 학점단위
double avrGrade;         // 학점평균 (계산값)
char *inputName;         // 사용자가 입력한 과목명

void Input_Grade()
        double inputGrade;      // 성적으로 받은 학점
void Read_FGrade()
void Cal_Grade()
void Show_Grade()
```

Key name	Data type	NULL	Default value	Key information
fGrade	int	X		학점단위
avrGrade	double	X	1	평점
sumGrade	double	X	0	학점의 합
grade	double	X		학점
inputName	char *	X		사용자가 입력한 과목명
inputGrade	double	X		사용자가 입력한 학점

DataBase Schema

▶ Class Date #1

```
int *inputDate;          // 사용자가 조회하거나 변경할 날로 입력한 날짜
char *inputSubject;      //      "       "       "       "       과목명

void Set_StartDate()
        int startDate;           // 한 학기가 시작되는 날짜
void Set_Date()
void Set_Subject()
void Show_Schedule()
void Add_Schedule()
        char *add_Sub;           // 정보가 추가될 과목의 과목명
        char *add_Info;          // 추가할 정보
void Add_ASubject()
        char *add_Sub;           // 보강하는 과목의 과목명
        int startTime;           // 보강이 시작하는 시간
        int endTime;             // 보강이 끝나는 시간
void Del_ASubject()
        char *del_Sub;           // 휴강되는 과목의 과목명
void Extend_ASubject()
        char *ex_Sub;            // 연강되는 과목의 과목명
        int endTime;             // 연강이 끝나는 시간
void Check_Attend()
        char *Sub;               // 과목명
        bool attend;             // 출석여부
```

DataBase Schema

▶ Class Date #2

Key name	Data type	NULL	Default value	Key information
inputDate	int	X		사용자가 입력한 날짜
inputDate	char *	X		사용자가 입력한 과목명
startDate	int	X		학 학기의 시작날짜
add_Sub	char *	X		정보를 추가할 과목명
add_Info	char *	X		추가할 정보
startTime	int	X		보강 시작시간
endTime	int	X		보강 및 연강 종료시간
del_Sub	char *	X		휴강 과목명
ex_Sub	char *	X		연강 과목명
Sub	char *	X		출석체크할 과목명
attend	bool	X	true / false	출석여부

DataBase Schema

▶ Class Link

void Show_Library()	//도서관 상황 웹페이지 링크오픈
void Show_Facilities	//교내시설 전화번호 웹페이지 링크오픈

Key name	Data type	NULL	Default value	Key information
X	X	X		X

감사합니다 :D